

eISBN: 978-1-60805-095-6

ISBN: 978-1-60805-425-1

Design of Analog Circuits through Symbolic Analysis

Editor:

Mourad Fakhfakh
University of Sfax
Tunisia

Co-Editors:

Esteban Tlelo-Cuautle
National Institute for Astrophysics, Optics and Electronics
Mexico

Francisco V. Fernández
IMSE-CNM, CSIC and University of Sevilla
Spain

Bentham  Books

eBooks End User License Agreement

Please read this license agreement carefully before using this eBook. Your use of this eBook/chapter constitutes your agreement to the terms and conditions set forth in this License Agreement. Bentham Science Publishers agrees to grant the user of this eBook/chapter, a non-exclusive, nontransferable license to download and use this eBook/chapter under the following terms and conditions:

1. This eBook/chapter may be downloaded and used by one user on one computer. The user may make one back-up copy of this publication to avoid losing it. The user may not give copies of this publication to others, or make it available for others to copy or download. For a multi-user license contact permission@benthamscience.org
2. All rights reserved: All content in this publication is copyrighted and Bentham Science Publishers own the copyright. You may not copy, reproduce, modify, remove, delete, augment, add to, publish, transmit, sell, resell, create derivative works from, or in any way exploit any of this publication's content, in any form by any means, in whole or in part, without the prior written permission from Bentham Science Publishers.
3. The user may print one or more copies/pages of this eBook/chapter for their personal use. The user may not print pages from this eBook/chapter or the entire printed eBook/chapter for general distribution, for promotion, for creating new works, or for resale. Specific permission must be obtained from the publisher for such requirements. Requests must be sent to the permissions department at E-mail: permission@benthamscience.org
4. The unauthorized use or distribution of copyrighted or other proprietary content is illegal and could subject the purchaser to substantial money damages. The purchaser will be liable for any damage resulting from misuse of this publication or any violation of this License Agreement, including any infringement of copyrights or proprietary rights.

Warranty Disclaimer: The publisher does not guarantee that the information in this publication is error-free, or warrants that it will meet the users' requirements or that the operation of the publication will be uninterrupted or error-free. This publication is provided "as is" without warranty of any kind, either express or implied or statutory, including, without limitation, implied warranties of merchantability and fitness for a particular purpose. The entire risk as to the results and performance of this publication is assumed by the user. In no event will the publisher be liable for any damages, including, without limitation, incidental and consequential damages and damages for lost data or profits arising out of the use or inability to use the publication. The entire liability of the publisher shall be limited to the amount actually paid by the user for the eBook or eBook license agreement.

Limitation of Liability: Under no circumstances shall Bentham Science Publishers, its staff, editors and authors, be liable for any special or consequential damages that result from the use of, or the inability to use, the materials in this site.

eBook Product Disclaimer: No responsibility is assumed by Bentham Science Publishers, its staff or members of the editorial board for any injury and/or damage to persons or property as a matter of products liability, negligence or otherwise, or from any use or operation of any methods, products instruction, advertisements or ideas contained in the publication purchased or read by the user(s). Any dispute will be governed exclusively by the laws of the U.A.E. and will be settled exclusively by the competent Court at the city of Dubai, U.A.E.

You (the user) acknowledge that you have read this Agreement, and agree to be bound by its terms and conditions.

Permission for Use of Material and Reproduction

Photocopying Information for Users Outside the USA: Bentham Science Publishers grants authorization for individuals to photocopy copyright material for private research use, on the sole basis that requests for such use are referred directly to the requestor's local Reproduction Rights Organization (RRO). The copyright fee is US \$25.00 per copy per article exclusive of any charge or fee levied. In order to contact your local RRO, please contact the International Federation of Reproduction Rights Organisations (IFRRO), Rue du Prince Royal 87, B-1050 Brussels, Belgium; Tel: +32 2 551 08 99; Fax: +32 2 551 08 95; E-mail: secretariat@ifrro.org; url: www.ifrro.org This authorization does not extend to any other kind of copying by any means, in any form, and for any purpose other than private research use.

Photocopying Information for Users in the USA: Authorization to photocopy items for internal or personal use, or the internal or personal use of specific clients, is granted by Bentham Science Publishers for libraries and other users registered with the Copyright Clearance Center (CCC) Transactional Reporting Services, provided that the appropriate fee of US \$25.00 per copy per chapter is paid directly to Copyright Clearance Center, 222 Rosewood Drive, Danvers MA 01923, USA. Refer also to www.copyright.com

CONTENTS

<i>About the Editors</i>	<i>i</i>
<i>Foreword</i>	<i>iii</i>
<i>Preface</i>	<i>v</i>
<i>List of Contributors</i>	<i>ix</i>

CHAPTERS

1. Recent Development in Symbolic Analysis: An Overview	3
<i>Sheldon X.-D. Tan and Esteban Tlelo-Cuautle</i>	
2. Modified Nodal Analysis	19
<i>Dalibor Birolek and Viera Biolkova</i>	
3. Modeling Active Devices with Nullor for Analog Signal Processing	61
<i>Carlos Sánchez-López</i>	
4. Generation of the Transfer Functions for MIMO Systems	83
<i>Mihai Iordache and Lucia Dumitriu</i>	
5. Symbolic Analysis of Analog Circuits by Flow-Graphs	115
<i>Mourad Fakhfakh, Irina Asenova and Mourad Loulou</i>	
6. Analysis and Synthesis of Electronic Circuits by Two-Graph Method	147
<i>Marian Pierzchala and Benedykt Rodanski</i>	
7. Approximation Techniques in Symbolic Circuit Analysis	173
<i>Francisco V. Fernández, Carlos Sánchez-López, Rafael Castro-López and Elisenda Roca-Moreno</i>	
8. Symbolic Analysis by Determinant Decision Diagrams and Applications	203
<i>Sheldon X.-D. Tan</i>	
9. Sensitivity Computation Based on the Auxiliary Circuits	229
<i>Lucia Dumitriu and Mihai Iordache</i>	
10. Symbolic Noise Analysis in Analog Circuits	265
<i>Carlos Sánchez-López</i>	
11. Symbolic Pole/Zero Analysis	287
<i>Francisco V. Fernández, Carlos Sánchez-López, Rafael Castro-López and Elisenda Roca-Moreno</i>	

- 12. Automatic Nonlinear Behavioral Model Generation Using Symbolic Circuit Analysis** 305
Ralf Sommer, Eckhard Hennig, Gregor Nitsche, Jochen Broz and Peter Schwarz
- 13. Nonlinear Template-Free Symbolic Performance Modeling for Design and Process Variation Analysis of Analog Circuits** 343
Trent McConaghy and Georges G.E. Gielen
- 14. Symbolic Analysis Techniques for Fault Diagnosis and Automatic Design of Analog Circuits** 361
Francesco Grasso, Antonio Luchetta, Stefano Manetti, Maria Cristina Piccirilli and Alberto Reatti
- 15. Symbolic Characterization of VCOs and its Application to Optimization Based Design** 399
Maria Helena Fino and Fernando V. Coito
- 16. AMS Synthesis Using Symbolic Methods** 413
Mauro Santos and Nuno Horta
- 17. Application of Symbolic Circuit Analysis for Failure Detection and Optimization of Industrial Integrated Circuits** 445
Ralf Sommer, Dominik Krauß, Eric Schäfer and Eckhard Hennig

About the Editors

Mourad Fakhfakh got the engineering and the PhD degrees from the national engineering school of Sfax Tunisia in 1996 and 2006 respectively. From 1998 to 2004 he worked in the Tunisian National Company of Electricity and Gas (STEG) as the head of the technical intervention department. In September 2004, he joined the higher institute of electronics and communications (ISECS) where he is currently working as an Assistant Professor. Dr. Fakhfakh serves as a reviewer in some prestigious journals and international conferences. He has been a TPC member in prominent international conferences and (co-)authored more than 80 papers in international journals and conferences. His research interests include analog & RF design automation, symbolic analysis, analog circuit synthesis and optimization techniques.

Esteban Tlelo-Cuautle received a B.Sc. degree from Instituto Tecnológico de Puebla (ITP), México in 1993. He then received both M.Sc. and Ph.D. degrees from Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE), México, in 1995 and 2000, respectively. From 1995-2000, he was a part of the electronic engineering department at ITP, and in 2001 he was appointed as full professor-researcher at INAOE. From 2009-2010, he served as a Visiting Researcher in the department of electrical engineering at the University of California Riverside. He has authored and co-authored four books, ten book chapters, 45 journal articles and around 100 conference papers. He is an IEEE Senior Member, IEICE Member, and a member of the National System for Researchers (SNI-CONACyT in México). He regularly serves as a reviewer in 18 high impact-factor journals in engineering and 15 internationally recognized conferences. He has been a member of Program Committees in prestigious international conferences. His research interests include systematic synthesis and behavioral modeling and simulation of linear and nonlinear circuits and systems, chaotic oscillators, symbolic analysis, multi-objective evolutionary algorithms, and analog/RF and mixed-signal design automation tools.

Francisco V. Fernández got the Physics-Electronics degree from the University of Seville in 1988 and his PhD degree in 1992. In 1993, he worked as a postdoctoral research fellow at Katholieke Universiteit Leuven (Belgium). Between 1995 and 2009 he was Associate Professor at the Dept. of Electronics and Electromagnetism of University of Seville, and since 2009 he is Full Professor at the same University. He is also a researcher at CSIC-IMSE-CNM. His research interests lie in the design and design methodologies of analog and mixed-signal circuits. Dr. Fernández has authored or edited three books and co-authored more than 100 papers in international journals and conferences. Dr. Fernández is currently the Editor-in-Chief of Integration, the VLSI Journal (Elsevier). He regularly serves at the Program Committee of several international conferences. Besides the above he has also participated as main researcher in several National and European R&D projects.

FOREWORD

Since the invention of the transistor and later on the integrated circuit, microelectronics has proven to be crucial as driver for innovation and welfare in our society. Many applications would not exist without integrated electronics. The computer, the digital media revolution (*e.g.* CD, DVD, mp3), the wireless communications, the internet, *etc.* would not exist without the relentless scaling progress and resulting performance improvement and cost reduction realized by microelectronics. In the near future this progress will continue even further with electronics contributing to reducing our energy consumption, enabling ubiquitous wireless monitoring and data streaming, invading daily objects networked in the internet of things, and even entering the human body.

In most of these applications the electronic systems interface with our physical world. Although the strength of electronics lies in the power of the digital compute engines that perform the signal and data processing, analog and mixed-signal circuits (or at higher frequencies RF circuits) are needed in the interfacing with the real-world signals. The current sub-100nm CMOS technologies even enable the integration of entire systems, including both the digital core and the analog interfaces, onto a single die (System on Chip) or into a single package (System in Package), offering even further cost and performance advantages. The inescapable use of analog circuits however poses some serious challenges. First of all, since the information is represented by continuous-valued signals, analog circuits are intrinsically vulnerable to all kinds of “interferences”, be it noise, process variability, crosstalk, *etc.* Secondly, the functionality and performance of analog circuits are a result of the complex interplay of many devices, the way how they are interconnected and their sizing and biasing. This makes the understanding of analog circuits and how they work quite complicated, and requires years from students and designers to develop the necessary insight and expertise. To assist them in this process, they mainly utilize the numerical circuit simulator SPICE (or any of its commercial implementations), which is indeed a great tool for design validation.

Symbolic analysis techniques offer a complementary way to analyze analog integrated circuits. Since the results of symbolic analysis are analytic equations that explicitly describe the functionality (*e.g.* transfer function, impedance, pole/zero...) of the circuit as a symbolic expression of the design variables, it lends itself more for designers to gain insight into the behavior of the circuit. Questions like “how does the circuit actually work?”, “at what node is the dominant pole located?”, “what is the relation between the bias current and the gain-bandwidth?”, “what is the impact of device mismatch on the PSRR of this circuit”, “which transistor(s) actually is the dominant noise source?”, “which nonlinearity is creating the most third-order distortion?” *etc.* can be answered by means of symbolic analysis. Originally hampered by the exponential complexity of the symbolic analysis problem, approximation techniques that typically exploit designer intent or common design practice had to be developed to make the results interpretable for humans. Symbolic expressions are however also useful in many other applications that require the repeated evaluation of circuit characteristics for a wide range of parameter values, such as for example encountered in circuit optimization or in yield analysis. Not requiring interpretation by humans, a wide range of efficient algorithmic techniques

has been developed in the past years to overcome the complexity issue and to make symbolic analysis tractable for circuits of practical size.

This edited book provides an overview of the current state of the art in symbolic analysis. The editors have compiled chapters from the key contributors to the field of symbolic analysis. These chapters neatly describe the latest results in terms of algorithms as well as applications of symbolic analysis techniques for analog circuits. Recent algorithmic improvements highlight the potential of today's symbolic analysis methods, both in terms of circuit complexity and of circuit characteristics that can be analyzed. The second part of the book presents the wide span of applications that utilize symbolic analysis, ranging from behavioral and performance modeling over design centering and fault diagnosis to automated design and system architectural exploration. These chapters clearly demonstrate the potential of symbolic analysis for analog circuits, in complement to or in combination with numerical simulation techniques.

This combination of state of the art information about algorithms and applications makes this book highly recommended reading for everyone interested in using symbolic analysis towards electronic circuit design, be it to improve his/her insight in circuits or for any of the other applications. The many circuit examples used throughout the whole book nicely illustrate the capabilities, and both circuit designers and CAD professionals will benefit from this information.

Please enjoy reading this book.

Prof. Georges G.E. Gielen
Katholieke Universiteit Leuven
Belgium

PREFACE

Symbolic analyzers have the potential to offer knowledge to sophomores as well as practitioners of analog circuit design. Actually, they are an essential complement to numerical simulators, since they provide insight into circuit behavior that numerical analyzers do not.

Symbolic analysis of electronic circuits addresses the generation of symbolic expressions for the parameters that describe the performance of linear and nonlinear circuits in the three domains: DC, AC and time; some or all the circuit parameters can be kept as symbols. In particular, in AC analysis these expressions are generated as ratios of polynomials in the *Laplace* variable s , with, as coefficients, the sum of products of small signal elements of the design vector. Symbolic expressions for characteristics, such as voltage and current gain, input and output impedance *etc.* can be easily obtained. Furthermore, it is easy to acquire network characteristics such as sensitivity and noise.

Due to the fact that these expressions remain valid during the change of component values (as long as models remain valid), designers can not only have an insight into the behavior of the network but also they can use the expressions to optimize the circuit's performances.

The first generation of symbolic analyzers was proposed in the late 60's. Since then, these analyzers were promptly recognized as crucial tools to automatically generate the behavioral model of analog integrated circuits. Actually, many techniques and programs have been presented for the symbolic analysis of linear and even nonlinear lumped time invariant circuits. These proposed techniques can be classified into four major categories:

- Matrix manipulation methods,
- Graph based approaches:
 - Tree enumeration methods,
 - (Signal) flow-graphs,
- Parameter extraction methods,
- Interpolation approaches.

This book presents details and exemplifies such famous techniques. This makes the book a good resource for circuit analysis. Thus, it is intended to students and researchers as well as for industry designers.

For large (and medium) size circuits, computed analytical expressions become too large and their number of terms is so important that its manipulation and interpretation become impossible. The book puts also the stress on this problem and highlights some techniques adapted to approximate symbolic description of circuit characteristics.

Moreover, industrial current R&D topics, recent developments and future trends in the field of symbolic analysis are highlighted.

The outline of the book is as follows:

Chapter 1, an overview chapter, presents a survey of the state-of-the art of symbolic analysis techniques for the design and the verification of analog integrated circuits. It highlights major developments in this field over the past several years and presents the outstanding problems for future research.

Chapter 2 gives a description of the Modified Nodal Analysis method. Analysis procedures of circuits containing classical and modern circuit elements are described. These methods are explained clearly in a number of examples with a view to the matrix form which is appropriate for computer implementation.

Chapter 3 describes the modeling of nullor-based active devices from the circuit level of abstraction: active devices (voltage-mode, current-mode and mixed-mode operations). Several examples using nullor-based models illustrate its use to calculate fully-symbolic small-signal characteristics of linear analog circuits as well as on the analysis of nonlinear circuits.

Chapter 4 presents two approaches based on the state equations and on the semi-state equations, respectively-for the generation of the transfer functions of MIMO systems in symbolic/numeric-symbolic matrix form. The state variable approach is developed in two variants: one uses the circuit matrices A, B, C, D , and the other one directly uses the state equations in the frequency domain. The method based on semi-state equations is developed both in two matrix representation and in a single matrix representation. Illustrative examples are given.

Chapter 5 details and exemplifies the flow-graph approach. It presents used approaches for the determination of the modified Coates flow-graphs. Symbolic sensitivity using this technique is briefly introduced.

Chapter 6 introduces the two-graph method. It details the construction of the corresponding networks, and presents such analysis both the frequency and time domains. The chapter also focuses on partitioning techniques and on active RC-circuit synthesis using the two-graph method.

Chapter 7 details approximation techniques in symbolic circuit analysis. It reviews from the first approximation techniques, only intended to improve the interpretability of symbolic results, to the most modern approximate symbolic analysis techniques based on the approximation of the network equations and the direct generation of the approximated symbolic results.

Chapter 8 deals with the symbolic analysis by the determinant decision diagrams. It shows how DDD-based symbolic analysis enables the exact symbolic analysis of many analog circuits substantially larger than the previous methods and open new applications for symbolic analysis. This chapter also covers approximation methods based on DDD-based symbolic analysis.

Chapter 9 deals with sensitivity computation based on the auxiliary circuits. Three procedures, for the sensitivity analysis using auxiliary circuits: the *Bykhovsky Perkins Cruz's* method, the incremental-circuit approach and the adjoint-circuit approach, are presented. Advantages and drawbacks of these procedures are pointed out. Some illustrative examples are exposed.

Chapter 10 focuses on symbolic noise analysis in analog circuits. The symbolic noise analysis of linear or linearized analog circuit at the transistor level of

abstraction is presented. A brief exposition on the signal path approach into analog circuits working in voltage-mode and current-mode is given. Computation of symbolic noise parameters of analog circuits is detailed. Two examples are introduced to illustrate the potentiality of the approach.

Chapter 11 treats symbolic pole/zero analysis. It reviews some recently reported techniques purposely developed to generate approximate symbolic expressions for poles and zeros under given accuracy constraints.

Chapter 12 focuses on the automatic nonlinear behavioral model generation using symbolic circuit analysis. It gives an overview of the symbolic/numerical algorithms for extraction of dominant behavior of linear systems, *e.g.* formulas for poles and zeros as well as algorithms for generating behavioral models from nonlinear differential-algebraic systems of equations. Applications of analysis and modeling of mixed electrical and mechanical systems are also presented.

Chapter 13 deals with nonlinear template-free symbolic performance models of design and process variation. It presents a method for generating performance models which need no prior specification of an equation template, and that handle strongly nonlinear circuits, statistical process variations, and a variety of analysis types.

Chapter 14 is concerned with symbolic analysis techniques for analog circuit fault diagnosis and automatic design. It presents a symbolic approach to the design centering problem, and details testability and fault diagnosis of analog integrated circuits. In addition, it highlights modeling of power electronic devices based on symbolic techniques.

Chapter 15 presents a brief description of CMOS ring VCOs, where particular emphasis for differential delay cell ring VCOs is given. Then, symbolic characterizations of the VCOs are detailed. The proposed approach relies on the resolution of the differential equations modeling each VCO delay cell. A working example for the case for a symmetric load ring VCO is presented, and obtained results are compared against those obtained with numerical simulation.

Chapter 16 describes the use of symbolic methods applied to the automatic exploration and characterization of analog and mixed-signal systems topologies and architectures. Synthesis techniques based on algorithm-level, using an HDL description, and employing, both, a modified signal flow-graph approach and a pattern recognition technique are presented.

Chapter 17 deals with application of symbolic circuit analysis for failure detection and optimization of industrial integrated circuits. It demonstrates how symbolic analysis and approximation allows analyzing industrial analog building blocks systems which were considered to be symbolically unsolvable before. Besides circuit failure analysis and modeling, a novel methodology that provides a new application-specific compensation for achieving highest performance requirements, is given. The methodology is demonstrated on several industrial examples.

Mourad Fakhfakh
Esteban Tlelo-Cuautle
Francisco V. Fernandez

List of Contributors

Irina Asenova	Higher school of transport, Sofia, Bulgaria
Dalibor Biolek	University of Defense, Faculty of Military Technologies, Brno, Czech Republic
Viera Biolkova	Brno University of Technology, Faculty of Electrical Engineering and Communication, Czech Republic
Jochen Broz	Private
Rafael Castro-Lopez	IMSE, CSIC and University of Sevilla, Spain
Fernando V. Coito	Instituto de Telecomunicações, Lisboa, Portugal
Lucia Dumitriu	Politehnica University of Bucharest, Romania
Mourad Fakhfakh	University of Sfax, Tunisia
Francisco V. Fernandez	IMSE, CSIC and university of Sevilla, Spain
Maria Helena Fino	Instituto de Telecomunicações, Lisboa, Portugal
Georges G.E. Gielen	K. U. Leuven, ESAT-MICAS, Leuven, Belgium
Francesco Grasso	Università degli Studi di Firenze, Italy
Eckhard Hennig	Institute for Microelectronic and Mechatronic Systems, Erfurt, Germany
Nuno Horta	Instituto de Telecomunicações, Lisboa, Portugal
Mihai Iordache	Politehnica University of Bucharest, Romania
Dominik Krauß	Ilmenau University of Technology, Germany
Mourad Loulou	University of Sfax, Tunisia
Antonio Luchetta	Università degli Studi di Firenze, Italy
Stefano Manetti	Università degli Studi di Firenze, Italy
Trent McConaghy	Solido Design Automation Inc., Canada (formerly with Katholieke Universiteit Leuven, Belgium)
Gregor Nitsche	Ilmenau University of Technology and Institute for Microelectronic and Mechatronic Systems GmbH, Germany
Maria Cristina Piccirilli	Università degli Studi di Firenze, Italy
Marian Pierzchala	University of Technology, Wrocław, Poland
Alberto Reatti	Università degli Studi di Firenze, Italy
Elisenda Roca-Moreno	IMSE, CSIC and University of Sevilla, Spain

Benedykt Rodanski	University of Technology, Sydney, Australia
Carlos Sánchez-Lopez	Autonomous University of Tlaxcala, México and IMSE-CNM, CSIC and University of Sevilla, Spain
Mauro Santos	Instituto de Telecomunicações, Lisboa, Portugal
Eric Schäfer	Ilmenau University of Technology, Germany
Peter Schwarz	Private
Ralf Sommer	Institute for Microelectronic and Mechatronic Systems GmbH, Germany
Sheldon X. –D. Tan	University of California, Riverside, CA
Esteban Tlelo-Cuautle	National Institute for Astrophysics, Optics and Electronics, Mexico

CHAPTER 1**Recent Development in Symbolic Analysis: An Overview****Sheldon X.-D. Tan^{1,*} and Esteban Tlelo-Cuautle²**¹*Department of Electrical Engineering, University of California, Riverside, USA and* ²*Department of Electronics, INAOE, MEXICO*

Abstract: This chapter gives an overview of the state of the art on symbolic analysis techniques for modeling, synthesis, design and verification of analog integrated circuits. Symbolic analysis is to generate analytic expressions for circuit performances in terms of circuit component parameters and frequency variables. It complements very well the results from numerical analysis for analog circuit designs. Furthermore, symbolic analysis is very instrumental for circuit designers to gain insights into the circuit's behavior for generating compact and behavioral models suitable for circuit sizing and synthesis. It is important towards the automatic analog synthesis and optimization. The chapter presents major developments in this field over the past several years and concludes with the outstanding problems for future research.

Keywords: Symbolic nodal analysis, behavioral modeling, active device, nullor, simplification approaches, determinant decision diagram, sparse matrix, model order reduction, moment matching, balanced truncation.

1. Introduction

Symbolic analysis is a systematic approach to obtaining the knowledge of analog building blocks in an analytic form. It is an essential complement to numerical simulation. Research on symbolic analysis can be dated back to 19th century. Developments in this field gained real momentum in the 1950's when electric computers were introduced and used in circuit analysis [1, 2]. As summarized in [2], the first general-purpose circuit analysis programs emerged in early 1960s, where the basic concepts behind computer-aided design and analysis of analog circuits to formulate network equations by matrix algebraic topological techniques, were developed [3]. The major works during that time were based on six formulation schemes [1, 2]: nodal, state variable, hybrid, tableau, signal flow, and ports. Among them, the nodal analysis method was adopted for the development of SPICE [4], which has been proven to be very popular from early 1970s. Methods developed from the 1950's to the 1980's can be basically categorized as [1, 2, 5, 9]: (1) Tree enumeration methods, (2) signal flow graph methods, (3) parameter extraction methods, (4) interpolation approaches, and (5) matrix-determinant methods. The details of these methods can be found in [1, 5, 6, 10].

Various methods are proposed to solve the long-standing circuit-size problem. For instance, several methods have been accompanied with the development of three kinds of procedures to reduce or simplify the final symbolic expression, they are: Simplification Before the Generation (SBG), Simplification During Generation (SDG), and Simplification After Generation (SAG). Besides, symbolic model order reduction techniques become to be another kind of simplification very useful for VLSI circuits [7]. Furthermore, the strategies used in modern symbolic analyzers in general come in two categories: those based on hierarchical decompositions [11-13], and those based

*Address correspondence to Sheldon X.-D. Tan: Department of Electrical Engineering, University of California, Riverside, USA; E-mail: stan@ee.ucr.edu

on approximations [14-20]. It is interesting to add a hybrid approach between these two, based on hierarchical decomposition but that incorporates approximation throughout the hierarchy [21].

Hierarchical decompositions generate symbolic expressions in a nested form [11-13]. There are several methods such as topological analysis [12], network formulation [11], and determinant decision diagram based hierarchical analysis [13]. All these methods are based on the sequence-of-expressions concept to obtain transfer functions. Approximations discard insignificant terms based on the relative numerical magnitude of symbolic parameters and the frequency defined at some nominal design points or over some ranges. It can be performed before [22], during [14, 17] and after [5, 23] the generation of symbolic terms.

The importance and increasing interest for symbolic analysis have been demonstrated by the success of modern symbolic analyzers such as ASAP [24], ISAAC [5], SCAPP [11], SYNAP [25] and RAINIER [17] and recent graph-based symbolic analyzer, SCAD3 [26] for analog integrated circuits. The developed symbolic analysis techniques have been used for analog circuit synthesis, optimization, reliability analysis, noise and distortion analysis, fault diagnosis, and design centering [8, 27]. Besides, symbolic approximation combined with numerical model order reduction techniques show promises for VLSI interconnects compact modeling [7, 28-30].

From the development of ISAAC [31], many symbolic simulators have been developed [8]. In the following, we try to briefly survey some recent developments. We remark that symbolic analysis and the related field have a large body of literature. Some relevant publications that are not cited in this chapter will not diminish their contributions to this field.

2. Symbolic Analysis for Analog Circuits

2.1. Behavioral modeling for active devices

Modeling is a preliminary work or construction that serves as a plan from which a final product can be made. Modeling at the transistor level of abstraction in the integrated circuit industry has roots in the primitives found in the popular SPICE simulator for integrated circuit design [1, 4]. Although the models have evolved to increased accuracy, improvements in speed of simulation have been small without going to higher levels of abstraction [6]. In the case of analog circuits, the ideal or most abstract behavior of the operational amplifier can be modeled by using the nullor element, which has shown its usefulness in circuit analysis, synthesis and design [32]. The suitability of the nullor to generate symbolic behavioral models is demonstrated in [33, 34], so that it can also be used to model any known and new active device element [35]. In Chapter 3 is introduced the modeling of active devices using nullors.

Symbolic behavioral modeling also is quite useful to describe voltage-controlled oscillators [36], and switched-capacitor Sigma Delta modulators [37]. Modeling in time-domain has been introduced in [38] for analog circuits, and up to now, it seems to be an open research area in VLSI design [7, 28, 30]. Other modeling approaches have been introduced, namely: posynomial model generation [39], and pole-zero extraction [40], which are more amenable for amplifier circuit design and optimization.

2.2. Circuit formulation

The formulation of the system of equations in analog circuits can be done by applying the well-known Modified Nodal Analysis (MNA) [1, 5-10]. However, when the non-ideal effects can be neglected, the nullor becomes to be a good element to model the behavior of the circuit to formulate a compacted system of equations [41]. It can also be used to transform voltage-mode to current-mode circuits [42]. Using nullors [43, 44], one is able to formulate the system of equations by applying only Nodal Analysis (NA) [45, 46], because all non-NA-compatible elements can be modeled by nullors to be NA-compatible ones [47, 48].

2.2.1. Nullor-based symbolic circuit analysis

The nullor consists of a nullator and a norator [45]. The nullator is an element which does not allow current flow through it, and the voltage across its terminals is zero. The norator is an element for which an arbitrary voltage can exist across it and simultaneously an arbitrary current can flow through it.

In the NA formulation, the four controlled sources, the active devices, and the independent voltage sources are transformed to be NA-compatibles, as shown in [47].

Let's consider the active RC filter shown by **Fig. 1**, which has been transformed to its nullor equivalent circuit. It has 11 nodes. The MNA formulation generates one equation for each node plus one equation for each opamp, leading to a system of order 15. On the other hand, the NA formulation (using nullors) generates a system of order equal to the number of nodes, minus the number of nullors (nullator-norator pair), leading to a system of order 6, as shown by (1). The symbolic transfer function is given by (2).

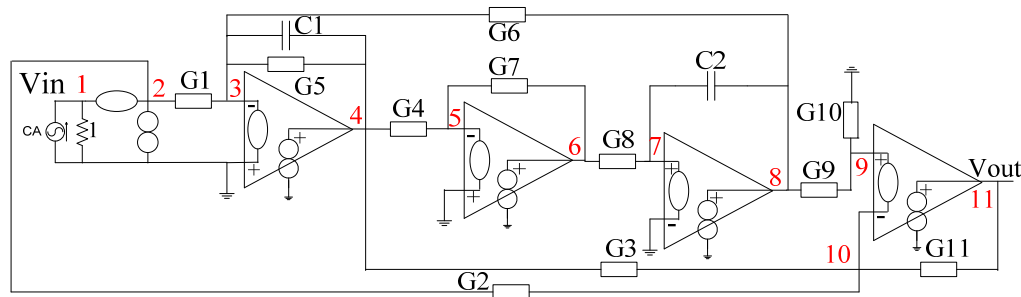


Figure 1: RC filter taken from page 955 of [49].

$$\begin{bmatrix} v_{in} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ -G1 & -G5 - sC1 & 0 & -G6 & 0 & 0 \\ 0 & -G4 & -G7 & 0 & 0 & 0 \\ 0 & 0 & -G8 & -sC2 & 0 & 0 \\ 0 & 0 & 0 & -G9 & G9 + G10 & 0 \\ -G2 & -G3 & 0 & 0 & G2 + G3 + G11 & -G11 \end{bmatrix} \begin{bmatrix} v_{1,2} \\ v_4 \\ v_6 \\ v_8 \\ v_{9,10} \\ v_{11} \end{bmatrix} \quad (1)$$

$$\frac{v_{out}}{v_{in}} = \frac{v_{11}}{v_{in}} = \quad (2)$$

$$\frac{-(G9 + G10)C1G2C2G7s^2 + ((G1G3 - G2G5)(G9 + G10))C2G7s - G4G8(G9G1(G2 + G3 + G11) + G2G6(G9 + G10))}{G11(G9 + G10)(G6G8G4 + sC2G7G5 + s^2C2G7C1)}$$

For the OTA filter shown by **Fig. 2**, the formulation generates the system given by (3), while the symbolic expression is given by (4).

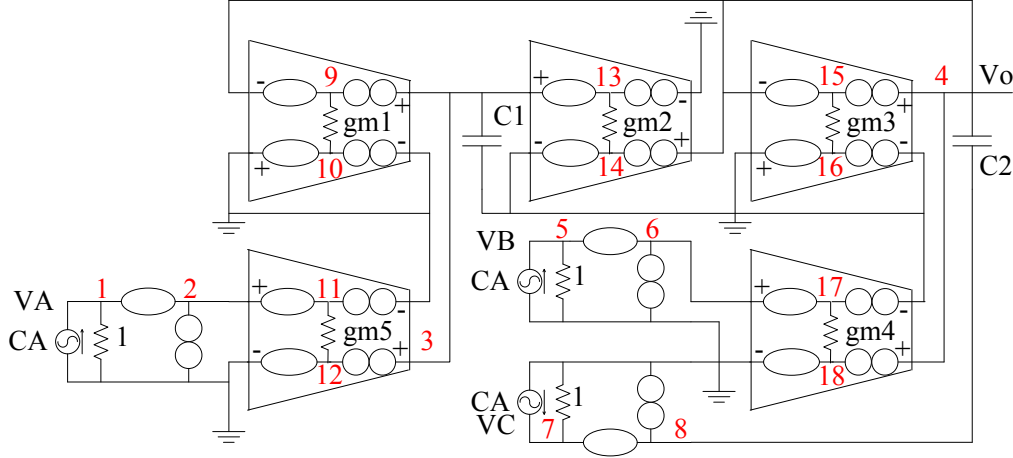


Figure 2: OTA filter taken from page 28 of [50].

$$\begin{bmatrix} v_A \\ 0 \\ 0 \\ v_B \\ v_C \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ -gm5 & sC1 & gm1 & 0 & 0 \\ 0 & -gm2 & sC2 + gm3 & -gm4 & -sC2 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_{1,2,11} \\ v_{3,13} \\ v_{4,9,15} \\ v_{5,6,17} \\ v_{7,8} \end{bmatrix} \quad (3)$$

$$v_o = v_4 = \frac{s^2 C_1 C_2 v_C + s C_1 g_{m4} v_B + g_{m2} g_{m5} v_A}{s^2 C_1 C_2 + s C_1 g_{m3} + g_{m1} g_{m2}} \quad (4)$$

From the examples given above, it can be appreciated the usefulness of the nullor element to model the abstract behavior and to apply the NA formulation for analog circuits. For transistor circuits including parasitic, the nullor based NA is developed in [34, 45].

Other formulation approaches can be found in [1, 2, 5-10]. Currently, new formulation methods are oriented to hybrid nonlinear circuits [51], state equations [7, 52], topological network [53], and to full custom circuits [54], which is oriented to compute delay models [28].

2.3. Determinant decision diagrams

One long-standing problem for symbolic analysis is the so-called circuit size problem: the number of symbolic terms generated can grow exponentially with the circuit size. This problem has been partially mitigated by a graph-based approach, called Determinant Decision Diagram (DDD) [55], where the symbolic terms are implicitly represented in a graph. Since the number of nodes in a graph is much smaller than the number of paths in a graph, DDDs can represent a huge number of symbolic terms from a determinant very efficiently and enable the exact symbolic analysis of much large analog circuits than all the existing approaches [7]. It has shown many advantages compared to conventional matrix-solution methods [1, 5-9, 56]. DDD-based symbolic analysis was further improved by logic operation DDD construction approach [57], and hierarchical analysis method [13, 58-60], for handling very large analog circuits. DDD-based symbolic analysis technique still remains the most efficient analysis method. Symbolic model order reduction *via* hierarchical approach, also called general Y-Delta transformation, was developed in [7, 59, 60].

The DDD method exploits the sparsity of the matrices for large-circuits. For small circuits, as the ones described by (1) and (3), it also improves the calculation of symbolic expressions. Some methods using other decision diagrams have also been proposed. For instance, [54] presents a symbolic timing analysis using algebraic decision diagrams (ADDs) to estimate delay. It analyzes delay with simple series-parallel reduction when possible and uses symbolic matrix techniques to handle complex circuit structures. In the time domain, the state variable method is adapted for efficient decomposition of large circuits [61]. The DDD method also shows advantages in regularity-based hierarchical symbolic analysis for large circuits [62]. In Chapter 8, a deeper description of the DDD is presented.

2.4. Noise and distortion analysis

Symbolic analysis has demonstrated its usefulness in computing second order effects such as noise and distortion. Some research has been presented in the area of integrated circuit design, as shown in [63-66]. The nature of the equations in noise analysis, allows applying DDDs [55] to improve the calculation of noise expressions. In Chapter 10, a symbolic noise analysis method for analog circuits is introduced.

The distortion analysis can be performed by symbolic analysis as shown by [41], where it is presented for bipolar transistor circuits. Up to now, the distortion analysis is generally performed for weakly nonlinearities [67], because the difficulty to generate analytical expressions in hard-distortion analysis [68]. For example, the application of symbolic analysis is suitable for the Dynamic Range optimization of continuous-time Gm-C filters [69], and the distortion analysis in single-, two- and three-stage amplifiers [70]. Combinations of symbolic methods with numerical analysis for nonlinear circuits are presented in [71, 72], also for weakly nonlinear circuits [73].

2.5. Symbolic approximation approaches

Approximation is to discard insignificant terms based on the relative numerical magnitude of symbolic parameters and the frequency defined at some nominal design points or over some ranges. It can be performed before [22], called Simplification Before The generation (SBG), during [14], called simplification during generation (SDG), and after [5, 23], called Simplification After Generation (SAG), the generation of symbolic terms [6].

Approximations after generation methods are the most reliable methods, but they require the expansion of product terms before approximations, and thus are limited to small analog circuits. Approximation during generation methods are based on the fact that product terms can be generated in a non-increasing order by finding the smallest weight spanning trees, by using matroid intersection algorithm or by finding the shortest paths in a graph. Approximations before generation method, removes circuit elements whose contributions to the transfer function containing them are negligible before product generation processes, and therefore it reduces the complexity of the final expanded expressions. For active transistor level circuits, the three approaches SBG, SDG and SAG, are useful to generate reduced analytical expressions [74].

The simplification approaches can be applied with the tree enumeration method, signal flow graph method, and matrix-determinant methods. Two recent symbolic approximation methods based on graph manipulations are presented in [75] and [76]. Also, reduction applied before generation has been used recently in [77-80]. In Chapter 7, a deeper description of some approximation techniques in symbolic circuit analysis is presented.

2.6. Application to circuit synthesis

The great variety of active devices used in analog signal processing applications [35], makes difficult the development of a canonical approach for circuit modeling and synthesis. However, it is possible to perform specific approaches, as the synthesis method shown in [81, 82], where it is introduced the generation, modeling, and analysis of current conveyor-based gyrators by performing symbolic operations.

As presented in [33], the current conveyor is an active device having three kinds of generations, two kinds of polarity and it can have multiple outputs. All these types or topologies for current conveyors can be designed by using four kinds of unity-gain cells: voltage and current followers and voltage and current mirrors. These four cells can be modeled by using nullors, so that the synthesis of the nullors can lead to multiple circuits performing the same behavior. In this manner, the synthesis approach presented in [81, 82], employs mirror elements and nullors to expand the admittance matrix describing the behavior of the current conveyor-based gyrator which is going to be synthesized. At the end of the symbolic expansion of the admittance matrix, the generalized impedance converter can be realized with a wide range of active elements, mainly by virtue of using mirror elements. This approach enhances the preliminary work introduced in [83]. Other applications of symbolic analysis to circuit synthesis can be found in [84-87], and in Chapters 6 and 16.

2.7. Miscellaneous applications

Undoubtedly, symbolic analysis is a powerful method very suitable to help to almost all stages and levels in electronic design of integrated circuits and systems. During the last decade, many works have been presented as suitable approaches competing with the state-of-the-art. This subsection only lists few works. For instance, symbolic analysis has been applied in circuit optimization at the layout level of description [88]. At circuit level of description, symbolic analysis has been applied in the following areas: fault diagnosis [89], design centering [90], and circuit reliability [91], where the authors propose the use of binary decision diagrams and algebraic decision diagrams for nanoscale circuits. Sensitivity analysis is also an open research problem [92], which is very much needed in model order reduction methods [7, 29, 30, 93], and in general, in the industrial analog IC design [94].

3. Symbolic analysis and model order reduction

A different approach for building compact models, especially for interconnects circuits modeled as RC/RLC circuits, is by means of Model Order Reduction (MOR) techniques [28, 95]. MOR is typically regarded as a purely numerical technique. However, MOR is actually a special symbolic analysis technique where the frequency variable is the only symbol kept. Actually, the increasing interests in parameterized or variational model order reduction methods, where one or more circuit parameters are kept as variables, make the distinctions between MOR and the traditional symbolic analysis to be blurred. Hence, the leveraging of the existing symbolic analysis techniques for variational MOR is attracting new research efforts.

3.1. Krylov subspace based reduction

The Krylov subspace method or moment-matching based approaches are popular MOR methods due to their efficiency and numerical robustness [96-102].

The Asymptotic Waveform Evaluation (AWE) method [96] first introduces the explicit moment-matching technique for fast interconnect modeling (mainly delay calculation). But AWE suffers from numerical instability owing to explicit moment-matching. To mitigate this problem, Krylov subspace based methods were proposed [97, 98], where implicit moment-matching is performed in a projection framework. Furthermore, to ensure the stability of the simulation process, PRIMA [99] was developed based on the Arnoldi process. PRIMA exploits the positive semi-definiteness of matrices in MNA formulation so that passivity can be easily preserved *via* congruency transformation [103]. More recently, SPRIM [101] further exploits the block structure of RLC formulation such that, in addition to passivity, structure information inherent to RLC circuits can be also preserved. At the same time, second-order moment-matching based approaches have been successfully developed, such as ENOR [100] to SAPOR [102].

3.2. Truncated balanced realization based reduction

While suitable for reduction of large-scale circuits, these techniques do not necessarily generate models as compact as desired [104]. Therefore, another approach, Truncated Balanced Realization (TBR), or balanced truncation, which was developed in the control community [105-109], has been studied intensively for interconnect modeling [110-119]. Standard balanced truncation methods, however, are known to be too expensive for direct application to large integrated circuit problems, owing to the cubic cost of solving two Lyapunov equations. In addition, it takes considerable knowledge of control theory and numerical procedures to implement balanced truncation in a stable way [120, 121]. Especially for nonstandard systems, additive decompositions and special treatments are causally needed [112, 122, 123]. To remedy this problem, several gramian approximation methods have been proposed [116, 118, 119, 124], where the approximated dominant subspace of a gramian can be obtained in a very efficient way. However, no rigorous error bounds exist for gramian approximation methods. The Single Gramian Approximation (SGA) technique (also called Poor Man's TBR or PMTBR) [116] was first proposed to reduce the system by projecting onto the approximated dominant subspace of the controllability gramian. This method works well for RC circuits, which can be naturally formulated in a first-order form with matrices both symmetric and positive-definite. However, for general RLCK (K is the coupling inductance) circuits, which models the on-chip global interconnects with fast signals, the first-order formulation

could be either symmetric or positive-definite, but not both. Therefore, preserving high accuracy and passivity cannot be achieved at the same time. There are several methods proposed to mitigate this problem. One of them, SBPOR [117], is based on the second-order formulation, which is both symmetric and positive-definite for RLCK interconnect circuits. In SBPOR, second-order gramians are defined based on a symmetric first-order realization.

As a result, both second-order gramians, which are also the leading blocks of the gramians of first-order realization, become the same and can be simultaneously diagonalized by a congruence transformation. As a result, it achieves passivity without sacrificing accuracy (it still approximates both controllability and observability gramians). Further, a fast SBPOR method, called SOGA, was proposed [119]. It computes the approximate gramians of one second-order formation from SBPOR to make the algorithm more computationally efficient.

3.3. Node elimination based reduction

Another quite different approach to circuit complexity reduction is by means of local node elimination and realization [125-129]. These types of methods are more close to the traditional hierarchical symbolic analysis. The major advantage of these methods over projection-based methods is that the reduction can be done in a local manner and no overall solution of the entire circuit is required and reduced models can be easily realized using RLCM (M denotes mutual inductance) elements. This idea was first explored by selective node elimination for RC circuits [125, 126], where time-constant analysis is used to select nodes for elimination. Node Reduction For Magnetic Coupling Interconnect (RLCM) circuits has recently become an active research area. Generalized Y-Delta transformation [128], RLCK circuit crunching [127], and branch merging [129] have been developed based on nodal analysis (NA), where inductance becomes susceptance in the admittance matrix. Since mutual inductance is coupled *via* branch currents, to perform nodal reduction, an equivalent 6-susceptance NA model is introduced in [128] to reduce two coupling current variables and template matching *via* geometrical programming is used to realize the model order reduced admittances, but its accuracy depends heavily on the selection of templates and only 1-port realization has been reported. Meanwhile, RLCK circuit crunching and branch merging methods are first-order approximation based on the nodal time-constant analysis. A more general node-elimination algorithm based on the general s-domain hierarchical graph-based symbolic reduction technique [59, 60, 104, 130] was proposed, which deal with circuits with large number of ports more efficiently than the projection based reduction techniques. But that papers still focus on projection based reduction techniques.

3.4. Parameterized and variational reduction

More order reduction with more variable parameters is important for variational or statistical modeling of analog and interconnect circuits due to process variations [131, 132]. One way for the symbolic model order reduction is to treat all the devices with variable parameters as outside devices and their corresponding nodes become new terminals for the revised circuits and traditional MOR is performed on the netlist-changed circuit [29]. However, this approach will lead to more terminals and make the sequential reduction process more difficult as existing MOR techniques are not efficient for reduction for networks with massive terminals [28].

Variational Model Order Reduction (MOR) considering process parameter variations is a new emerging field, and some early approaches have been proposed. Existing

approaches consist of perturbation-based methods [133], first-order and Gaussian-only interconnect delay modeling method [134], multi-dimensional moment matching based methods [135, 136], interval analysis-based methods [137, 138], and variational subspace-based methods [139]. The perturbation based method [133] basically applies the perturbation theory to represent the matrix operations in an explicit variational form. This approach, however, only works for very small variations.

Multi-dimensional moment matching methods [135, 136] treat the random variables as the complex frequency variables, which results in so-called multi-dimensional moments in terms of both random variables and frequency variables. Those methods, however, suffer the exponential growth of moment terms with respect to the number of variables. The interval-valued MOR method, rather than performing the calculations of model order reduction on real-valued scalars, uses intervals on each line to approximate each statistical variation range [140]. An interval-valued MOR method based on affine arithmetic model was proposed recently [137, 138], where the pole and residues in transfer functions become interval-valued. But this interval method still suffers over-estimation problems, especially for computation intensive numerical operations like projection-based model order reduction algorithms.

Recently, statistical interconnect analysis methods using stochastic FEM method have been proposed for timing analysis [141-143]. As a result, we can convert a statistical problem into a deterministic one by using the Galerkin method. The orthogonal polynomial method can deal with different kinds of distributions (Gaussian, lognormal, uniform, *etc.*). However, existing approaches may result in very large augmented circuit matrices to solve after applying the Galerkin method. This problem is partially mitigated by using the explicit moment-matching method to compute delay distributions [142].

Another recently proposed statistical MOR method is based on the variational subspace concept (also called varPMTBR method) [139]. The varPMTBR method treats other variables as the frequency variable. Unlike multi-dimensional moment matching methods, varPMTBR computes Gramians by random sampling in both frequency variable and random variable space. The main benefit of this method is that the number of samplings in building the variation subspace can be much cheaper than that of normal MC samplings. However, this method is far from mature and many problems remain to be solved. For instance, how to select the best sampling set to minimize the computing costs and reduce the errors in the reduced models still remains an open problem.

4. Outstanding problems for symbolic analysis

Although symbolic analysis is not a new topic, many outstanding problems are still open and needs further research efforts [6-8].

In the following, we propose some potential problems:

1. Combine the traditional symbolic analysis techniques with numerical model order reduction techniques for variational and statistical compact modeling for interconnect and analog circuits considering process variations.
2. Although with the advent of DDDs, symbolic analysis capability has been significantly improved, but modern symbolic analyzers still can't handle very large analog circuits. New approximation techniques and symbolic model order reduction techniques are required [6, 7].

3. Compact modeling for general nonlinear circuits (weakly nonlinear and strong nonlinear) still remain an open problem. No systematic approaches yet to be proposed to solve this problem. Existing approach based on Volterra functions [144, 145] and piece-wise linear modeling approaches [72, 146] still can't compete with mainstream table and fitting based models for numerical analysis. Novel innovation in these areas is strongly needed. Also, compact modeling is needed in the behavioral modeling of new active devices [35, 80].

4. The problem of huge symbolic expressions can be solved through the development of new SBG, SDG, SAG and symbolic model order reduction. For the SBG approaches: Before the formulation of the system of equations, one can explore on the use of the nullor to discriminate non-dominant terms [33, 34, 45], to formulate a compacted system of equations. Also, one can explore on the application of graph approaches to reduce the number of symbolic elements [75, 76]. For the SAG approaches, when the system of equations or the graph has been solved by DDD, Boolean logic operations or graph methods, the solution can generate very huge symbolic expressions, research in this area is still needed, for instance expressions have been reduced by applying symbolic comparisons [74].

Other outstanding problems for further research can be found in the following chapters.

Acknowledgements

The authors acknowledge the support from UC-MEXUS-CONACYT collaboration grant CN-09-310, and to CONACyT for the sabbatical stay of Dr. E. Tlelo-Cuautle at University of California at Riverside during 2009-2010, and for the grant for the project 48396-Y.

References

- [1] L.O. Chua and P.M. Lin, *Computer-aided analysis of electronic circuits*, N.J.: Prentice Hall, 1975.
- [2] R.W. Jensen and L.P. McNamee, *Handbook of circuit analysis languages and techniques*. N. J.: Prentice Hall, 1976.
- [3] G. Kron, *Tensor analysis of networks*. New York: Wiley, 1939.
- [4] L.W. Nagel and D.O. Pederson, *SPICE (Simulation Program with Integrated Circuit Emphasis)*. Memorandum ERL-M382, Calif.: University of California, Electronics Research Laboratory: Berkeley, 1973.
- [5] G. Gielen and W. Sansen, *Symbolic analysis for automated design of analog integrated circuits*. USA: Kluwer Academic Publishers, 1991.
- [6] F.V. Fernández, A. Rodríguez-Vázquez, J.L. Huertas, and G.E. Gielen, *Symbolic analysis techniques: applications to analog design automation*. Piscataway, NJ: IEEE Press, 1998.
- [7] Z. Qin, S. X.-D. Tan, and C.-K. Cheng, *Symbolic analysis and reduction of VLSI circuits*. Boston M.A.: Kluwer Academic Publishers, 2005.
- [8] R.A. Rutenbar, G.E. Gielen, and B.A. Antao, *Computer-aided design of analog integrated circuits and systems*. NY: Wiley, 2002.
- [9] G. Gielen, P. Wambacq, and W. Sansen, "Symbolic analysis methods and applications for analog circuits: a tutorial overview," *Proceedings of the IEEE*, vol. 82, no. 2, pp. 287–304, 1994.
- [10] P.M. Lin, *Symbolic network analysis*. Elsevier Science Publishers B.V., 1991.
- [11] M.M. Hassoun and P.M. Lin, "A hierarchical network approach to symbolic analysis of large scale networks," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 42, pp. 201–211, April 1995.
- [12] J.A. Starzky and A. Konczykowska, "Flowgraph analysis of large electronic networks," *IEEE Transactions on Circuits and Systems*, vol. 33, pp. 302–315, March 1986.

- [13] S. X.-D. Tan and R. C.-J. Shi, "Hierarchical symbolic analysis of large analog circuits via determinant decision diagrams," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, vol. 19, pp. 401–412, April 2000.
- [14] F.V. Fernández, P. Wambacq, G. Gielen, A. Rodríguez-Vázquez, and W. Sansen, "Symbolic analysis of large analog integrated circuits by approximation during expression generation," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, 1994, pp. 25–28.
- [15] P. Wambacq, G. Gielen, and W. Sansen, "A cancellation-free algorithm for the symbolic simulation of large analog circuits," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, May 1992, pp. 1157–1160.
- [16] J.-J. Hsu and C. Sechen, "DC small signal symbolic analysis of large analog integrated circuits," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 41, pp. 817–828, Dec. 1994.
- [17] Q. Yu and C. Sechen, "A unified approach to the approximate symbolic analysis of large analog integrated circuits," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 43, pp. 656–669, Aug. 1996.
- [18] F. Leyn, G. Gielen, and W. Sansen, "Analog small-signal modeling – part I: behavioral signal path modeling for analog integrated circuits," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 48, no. 7, pp. 701–711, July 2001.
- [19] F. Leyn, G. Gielen, and W. Sansen, "Analog small-signal modeling – part II: elementary transistor stages analyzed with behavioral signal path modeling," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 48, no. 7, pp. 712–721, July 2001.
- [20] S. X.-D. Tan and C.-J. Shi, "Efficient approximation of symbolic expressions for analog behavioral modeling and analysis," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, vol. 23, pp. 907–918, June 2004.
- [21] O. Guerra, E. Roca, F.V. Fernández and A. Rodríguez-Vázquez, "Approximate symbolic analysis of hierarchically decomposed analog Circuits," *Analog Integrated Circuits and Signal Processing*, vol. 31, pp. 131-146, May 2002.
- [22] J. Rodríguez-García, O. Guerra, E. Roca, F.V. Fernández, and A. Rodríguez-Vázquez, "Error control in simplification before generation algorithms for symbolic analysis of large analogue circuits," *Electronics Letters*, vol. 35, no. 4, pp. 260–261, 1999.
- [23] S.J. Seda, M.G.R. Degrauwe, and W. Fichtner, "Lazy-expansion symbolic expression approximation in SYNAP," in *Proc. IEEE International Conference on Computer Aided Design (ICCAD)*, Nov. 1992, pp. 310–317.
- [24] F.V. Fernández, A. Rodríguez-Vázquez, and J.L. Huertas, "A tool for symbolic analysis of analog integrated circuits including pole/zero extraction," in *Proc. European Conf. Circuit Theory and Design (ECCTD)*, 1991, pp. 751–761.
- [25] S.J. Seda, M.G.R. Degrauwe, and W. Fichtner, "A symbolic analysis tool for analog circuit design automation," in *Proc. IEEE International Conference on Computer Aided Design (ICCAD)*, Nov. 1988, pp. 488–491.
- [26] S. X.-D. Tan, *Symbolic analysis of large analog circuits with determinant decision diagrams*. University of Iowa: Ph.D. Thesis, 1999.
- [27] G. Gielen and R. Rutenbar, "Computer-aided design of analog and mixed-signal integrated circuits," *Proceedings of the IEEE*, vol. 88, pp. 703–717, Dec. 2000.
- [28] S. X.-D. Tan and L. He, *Advanced model order reduction techniques in VLSI design*. UK: Cambridge University Press, 2007.
- [29] G. Shi, B. Hu, and C. J.-R. Shi, "On symbolic model order reduction," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, vol. 25, no. 7, pp. 1257–1272, 2006.
- [30] R. Sommer, T. Halfmann, and J. Broz, "Automated behavioral modeling and analytical model-order reduction by application of symbolic circuit analysis for multi-physical systems," *Simulation Modeling Practice and Theory*, vol. 16, no. 8, pp. 1024–1039, 2008.
- [31] G. Gielen, H. Walscharts, and W. Sansen, "ISAAC: a symbolic simulator for analog integrated circuits," *IEEE Journal of Solid-State Circuits*, vol. 24, no. 6, pp. 1587–1597, 1989.
- [32] P. Kumar and R. Senani, "Bibliography on nullors and their applications in circuit analysis, synthesis and design," *Analog Integrated Circuits and Signal Processing*, vol. 33, no. 1, pp. 65–76, 2002.
- [33] E. Tlelo-Cuautle, C. Sánchez-López, and D. Moro-Frías, "Symbolic analysis of (MO)(I)CCI(II)(III)-based analog circuits," *International Journal of Circuit Theory and Applications*, 2009. DOI: 10.1002/cta.582.

- [34] C. Sánchez-López and E. Tlelo-Cuautle, "Behavioral model generation of current-mode analog circuits," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, 2009, pp. 2761–2764.
- [35] D. Biolková, R. Senani, V. Biolková, and Z. Kolka, "Active elements for analog signal processing: classification, review, and new proposals," *Radioengineering*, vol. 17, no. 4, pp. 15–32, 2008.
- [36] Y. Zhao and Z.-G. Wang, "20-ghz differential colpitts VCO in 0.35-um BiCMOS," *Journal of Infrared, Millimeter and Terahertz Waves*, vol. 30, no. 3, pp. 250–258, 2009.
- [37] G. Suarez, M. Jimenez, and F.O. Fernández, "Behavioral modeling methods for switched-capacitor sigma delta modulators," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 54, no. 6, pp. 1236–1244, 2007.
- [38] R. Chakraborty, M. Ranjan, and R. Vemuri, "Symbolic time-domain behavioral and performance modeling of linear analog circuits using an efficient symbolic newton-iteration algorithm for pole extraction," in *Proc. International Conference on VLSI Design*, 2005, pp. 689–694.
- [39] T. Eeckelaert, W. Daems, G. Gielen, and W. Sansen, "Generalized simulation-based posynomial model generation for analog integrated circuits," *Analog Integrated Circuits and Signal Processing*, vol. 40, no. 3, pp. 193–203, 2004.
- [40] O. Guerra, J. Rodríguez-García, F.V. Fernández, and A. Rodríguez-Vázquez, "A symbolic pole/zero extraction methodology based on analysis of circuit time-constants," *Analog Integrated Circuits and Signal Processing*, vol. 31, no. 2, pp. 101–117, 2002.
- [41] H. Floberg, *Symbolic analysis in analog integrated circuit design*. USA: Kluwer Academic Publishers, 1997.
- [42] A. Carlosena and G. Moschytz, "Nullators and norators in voltage to current mode transformations," *International Journal of Circuit Theory and Applications*, vol. 21, no. 4, pp. 421–424, 1993.
- [43] J. Svoboda, "Current conveyors, operational amplifiers and nullors," *IEE Proceedings G Circuits, Devices and Systems*, vol. 136, no. 6, pp. 317–322, 1989.
- [44] J. Svoboda, "Using nullors to analyse linear networks," *International Journal of Circuit Theory and Applications*, vol. 14, no. 3, pp. 169–180, 1986.
- [45] C. Sánchez-López, D. Moro-Frias, and E. Tlelo-Cuautle, "Improving the formulation process of the system of equations of analog circuits," in *Proc. International Workshop on Symbolic and Numerical Methods, Modeling and Applications to Circuit Design (SM2ACD)*, 2008, pp. 102–106.
- [46] E. Tlelo-Cuautle, E. Martínez-Romero, C. Sánchez-López, and S. X.-D. Tan, "Symbolic formulation method for mixed-mode analog circuits using nullors," in *Proc. IEEE International Conference on Electronics, Circuits, and Systems (ICECS)*, 2009, pp. 856–859.
- [47] E. Tlelo-Cuautle, C. Sánchez-López, and F. Sandoval-Ibarra, "Computing symbolic expressions in analog circuits using nullors," *Computación y Sistemas*, vol. 9, no. 2, pp. 119–132, 2005. [Online] Available: <http://www.scielo.org.mx/pdf/cys/v9n2/v9n2a4.pdf>
- [48] E. Tlelo-Cuautle, C. Sánchez-López, and F. Sandoval-Ibarra, "Symbolic analysis: a formulation approach by manipulating data structures," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, vol. IV, 2003, pp. 640–643.
- [49] C. Toumazou, G. Moschytz, and B. Gilbert, *Trade-offs in analog circuit design*. Netherlands: Kluwer Academic Publishers, 2002.
- [50] L. Randall and E. Sánchez-Sinencio, "Active filter design using operational transconductance amplifiers: A tutorial," *IEEE Circuits and Devices Magazine*, pp. 20–32, 1985.
- [51] L. Dumitriu, M. Iordache, and N. Voicu, "Symbolic hybrid analysis of nonlinear analog circuits," in *Proc. IEEE European Conference Circuit Theory and Design (ECCTD)*, 2007, pp. 970–973.
- [52] M. Iordache and L. Dumitriu, "Multi-time method based on state equations for RF circuit analysis," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, 2007, pp. 517–520.
- [53] W. Chen and G. Shi, "Implementation of a symbolic circuit simulator for topological network analysis," in *Proc. IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, 2006, pp. 1368–1372.
- [54] H. Song, K. Nepal, R. Bahar, and J. Grodstein, "Timing analysis for full-custom circuits using symbolic dc formulations," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 9, pp. 1815–1830, 2006.
- [55] C. Shi and S. Tan, "Canonical symbolic analysis of large analog circuits with determinant decision diagrams," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, no. 1, pp. 1–18, 2000.

- [56] W. Verhaegen and G. Gielen, "Efficient DDD-based symbolic analysis of linear analog circuits," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 49, no. 7, pp. 474–487, 2002.
- [57] S. X.-D. Tan, "Symbolic analysis of analog circuits by boolean logic operations," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 53, no. 11, pp. 1313–1317, 2006.
- [58] S. X.-D. Tan, W. Guo, and Z. Qi, "Hierarchical approach to exact symbolic analysis of large analog circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 8, pp. 1241–1250, 2005.
- [59] S. X.-D. Tan, "A general s-domain hierarchical network reduction algorithm," in *Proc. IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, Nov. 2003, pp. 650–657.
- [60] S. X.-D. Tan, "A general hierarchical circuit modeling and simulation algorithm," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, pp. 418–434, April 2005.
- [61] M. Iordache and L. Dumitriu, "Efficient decomposition techniques for symbolic analysis of large-scale analog circuits by state variable method," *Analog Integrated Circuits and Signal Processing*, vol. 40, no. 3, pp. 235–253, 2004.
- [62] A. Doboli and R. Vemuri, "A regularity-based hierarchical symbolic analysis method for large-scale analog networks," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 48, no. 11, pp. 1054–1068, 2001.
- [63] C. Sánchez-López, E. Tlelo-Cuautle, M. Fakhfakh, and M. Loulou, "Computing simplified noise-symbolic-expressions in CMOS CCs by applying SPA and SAG," in *Proc. IEEE International Conference on Microelectronics (ICM)*, 2007, pp. 159–162.
- [64] C. Sánchez-López and E. Tlelo-Cuautle, "Symbolic noise analysis in gm-C filters," in *Proc. IEEE Electronics, Robotics and Automotive Mechanics Conference (CERMA)*, vol. I, 2006, pp. 49–53.
- [65] E. Tlelo-Cuautle and C. Sánchez-López, "Symbolic computation of NF of transistor circuits," *IEICE Transactions Fundamentals of Electronics, Communications and Computer Sciences*, vol. E87-A, no. 9, pp. 2420–2425, 2004.
- [66] C. Sánchez-López and E. Tlelo-Cuautle, "Symbolic noise analysis in analog integrated circuits," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, vol. V, 2004, pp. 245–248.
- [67] P. Li and L.T. Pileggi, "Compact reduced-order modeling of weakly nonlinear analog and RF circuits," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, vol. 23, no. 2, pp. 184–203, 2005.
- [68] P. Wambacq, G. Gielen, P. Kinget, and W. Sansen, "High-frequency distortion analysis of analog integrated circuits," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 46, no. 3, pp. 335–345, 1999.
- [69] J. Fernandez-Bootello, M. Delgado-Restituto, and A. Rodríguez-Vázquez, "Matrix methods for the dynamic range optimization of continuous-time g(m)-C filters," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 55, no. 9, pp. 2525–2538, 2008.
- [70] L. Hernes and W. Sansen, "Distortion in single-, two- and three-stage amplifiers," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 52, no. 5, pp. 846–856, 2005.
- [71] G. Rozakis and A. Samelis, "Symbolic/numerical nonlinear circuit analysis using volterra series," in *Proc. 36th European Microwave Conference*, 2006, pp. 1610–1613.
- [72] A. Manthe, L. Zhao, and C. Shi, "Symbolic analysis of analog circuits with hard nonlinearity," in *Proc. IEEE/ACM Design Automation Conference (DAC)*, 2003, pp. 542–545.
- [73] W. Daems, G. Gielen, and W. Sansen, "A fitting approach to generate symbolic expressions for linear and nonlinear analog circuit performance characteristics," in *Proc. Design, Automation and Test in Europe (DATE)*, 2002, pp. 268–273.
- [74] C. Sánchez-López and E. Tlelo-Cuautle, "Novel SBG, SDG and SAG techniques for symbolic analysis of analog integrated circuits," in *International Workshop on Symbolic and Numerical Methods, Modeling and Applications to Circuit Design (SM2ACD)*, 2008, pp. 17–22.
- [75] G. Shi, W. Chen, and C. Shi, "A graph reduction approach analysis," in *Proc. Asia South Pacific Design Automation Conference (ASPDAC)*, 2007, pp. 197–202.
- [76] Z. Kolka and M. Volkova, "Implementation of graph-based symbolic simplification," in *Proc. International Conference on Radioelektronika*, 2007, pp. 43–46.

- [77] S. Djordjevic and P. Petkovic, "Generation of factorized symbolic network function by circuit topology reduction," in *Proc. IEEE International Conference on Microelectronics (MIEL)*, 2004, pp. 773–776.
- [78] W. Daems, G. Gielen, and W. Sansen, "Circuit simplification for the symbolic analysis of analog integrated circuits," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, vol. 21, no. 4, pp. 395–407, 2002.
- [79] M. Pierzchala and B. Rodanski, "Generation of sequential symbolic network functions for large-scale networks by circuit reduction two-port," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 48, no. 7, pp. 906–909, 2001.
- [80] E. Tlelo-Cuautle, C. Sánchez-López, E. Martínez-Romero, and S. X.-D. Tan, "Symbolic analysis of analog circuits containing voltage mirrors and current mirrors," *Analog Integrated Circuits and Signal Processing*, 2010. DOI: 10.1007/s10470-010-9455-y
- [81] R. Saad and A. Soliman, "Generation, modeling, and analysis of CCII-based gyrators using the generalized symbolic framework for linear active circuits," *International Journal of Circuit Theory and Applications*, vol. 36, no. 3, pp. 289–309, 2008.
- [82] R. Saad and A. Soliman, "Use of mirror elements in the active device synthesis by admittance matrix expansion," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 55, no. 9, pp. 2726–2735, 2008.
- [83] R. Cabeza and A. Carlosena, "On the use of symbolic analyzers in circuit synthesis," *Analog Integrated Circuits and Signal Processing*, vol. 25, no. 1, pp. 67–75, 2000.
- [84] H. Yang and R. Vemuri, "Efficient temperature-dependent symbolic sensitivity analysis and symbolic performance evaluation in analog circuit synthesis," in *Proc. Design, Automation and Test in Europe (DATE)*, 2006, pp. 281–282.
- [85] H. Zhang and A. Doboli, "Fast time-domain symbolic simulation for synthesis of sigma-delta analog-digital converters," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, 2004, pp. 125–128.
- [86] M. Ranjan, A. Bhaduri, and W. Verhaegen, "Use of symbolic performance models in layout-inclusive RF low noise amplifier synthesis," in *Proc. IEEE International Workshop on Behavioral Modeling and Simulation (BMAS)*, 2004, pp. 130–134.
- [87] R. Castro-López, O. Guerra, F.V. Fernández, and A. Rodríguez-Vázquez, "Synthesis of a wireless communication analog back-end based on a mismatch-aware symbolic approach," *Analog Integrated Circuits and Signal Processing*, vol. 40, no. 3, pp. 215–233, 2004.
- [88] L. Zhang, N. Jangkrajarn, S. Bhattacharya, and C. Shi, "Parasitic-aware optimization and retargeting of analog layouts: A symbolic-template approach," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, vol. 27, no. 5, pp. 791–802, 2008.
- [89] F. Grasso, A. Luchetta, S. Manetti, and M. Piccirilli, "A method for the automatic selection of test frequencies in analog fault diagnosis," *IEEE Transactions on Instrumentation and Measurement*, vol. 56, no. 6, pp. 2322–2329, 2007.
- [90] F. Grasso, S. Manetti, and M. Piccirilli, "A symbolic approach to design centering of analog circuits," *Microelectronics Reliability*, vol. 47, no. 8, pp. 1288–1295, 2007.
- [91] N. Miskov-Zivanov and D. Marculescu, "Circuit reliability analysis using symbolic techniques," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, vol. 25, no. 12, pp. 2638–2649, 2006.
- [92] I. Asenova, "Symbolic sensitivity analysis using nullators, norators and modified coates signal - flow graph," in *Proc. International Symposium ELMAR*, 2008, pp. 245–248.
- [93] M. Iordache and L. Dumitriu, "Time domain diakoptic analysis based on reduced-order state equations," *International Journal of Bifurcation and Chaos*, vol. 17, no. 10, pp. 3625–3631, 2007.
- [94] R. Sommer and E. Hennig, "Application of symbolic analysis in the industrial analog IC design," *Modeling and Simulation*, pp. 666–673, 2002.
- [95] A.C. Antoulas, *Approximation of Large-Scale Dynamical Systems*. USA: The Society for Industrial and Applied Mathematics (SIAM), 2005.
- [96] L.T. Pillage and R.A. Rohrer, "Asymptotic waveform evaluation for timing analysis," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, pp. 352–366, April 1990.
- [97] P. Feldmann and R.W. Freund, "Efficient linear circuit analysis by Padé approximation via the lanczos process," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, pp. 639–649, 1995.
- [98] L.E.M. Silveira, M. Kamon, and J. White, "A coordinate-transformed Arnoldi algorithm for generating guaranteed stable reduced-order models of RLC circuits," in *Proc. IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, 1996, pp. 288–294.

- [99] A. Odabasioglu, M. Celik, and L. Pileggi, "PRIMA: Passive reduced-order interconnect macromodeling algorithm," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, pp. 645–654, 1998.
- [100] B.N. Sheehan, "ENOR: model order reduction of RLC circuits using nodal equations for efficient factorization," in *Proc. IEEE/ACM Design Automation Conference (DAC)*, 1999, pp. 17–21.
- [101] R.W. Freund, "SPRIM: structure-preserving reduced-order interconnect macromodeling," in *Proc. IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, 2004, pp. 80–87.
- [102] Y. Su, J. Wang, X. Zeng, Z. Bai, C. Chiang, and D. Zhou, "SAPOR: second-order Arnoldi method for passive order reduction of RCS circuits," in *Proc. IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, 2004, pp. 74–79.
- [103] K.J. Kerns and A.T. Yang, "Stable and efficient reduction of large, multiport RC network by pole analysis via congruence transformations," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, vol. 16, pp. 734–744, July 1998.
- [104] Z. Qi, H. Yu, P. Liu, S. X.-D. Tan, and L. He, "Wideband passive multi-port model order reduction and realization of RLCM circuits," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, vol. 25, pp. 1496–1509, Aug. 2006.
- [105] B. Moore, "Principal component analysis in linear systems: Controllability, and observability, and model reduction," *IEEE Transactions Automatic Control*, vol. 26, no. 1, pp. 17–32, 1981.
- [106] K. Glover, "All optimal Hankel-norm approximations of linear multi-variable systems and their L_∞ error bounds," *International Journal Control*, vol. 36, pp. 1115–1193, 1984.
- [107] U. Desai and D. Pal, "A transformation approach to stochastic model reduction," *IEEE Transactions Automatic Control*, vol. 29, pp. 1097–1100, 1984.
- [108] P. Harshavardhana, E. Jonckheere, and L. Silverman, "Stochastic balancing and approximation-stability and minimality," *IEEE Transactions Automatic Control*, vol. 29, pp. 744–746, 1984.
- [109] D.G. Meyer and S. Srinivasan, "Balancing and model reduction for second-order form linear systems," *IEEE Transactions Automatic Control*, vol. AC-41, pp. 1632–1644, 1996.
- [110] J.R. Li, F. Wang, and J. White, "An efficient Lyapunov equation-based approach for generating reduced-order models of interconnect," in *Proc. IEEE/ACM Design Automation Conference (DAC)*, 1999, pp. 1–6.
- [111] J.R. Li, *Model reduction of large linear systems via low rank system gramians*, Ph.D. Thesis. MIT, 2002.
- [112] J.R. Phillips, L. Daniel, and L.M. Silveira, "Guaranteed passive balancing transformation for model order reduction," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, vol. 22, no. 8, pp. 1027–1041, 2003.
- [113] N. Wang and V. Balakrishnan, "Fast balanced stochastic truncation via a quadratic extension of the alternating direction implicit iteration," in *Proc. IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, 2005, pp. 801–805.
- [114] D. Vasilyev and J. White, "A more reliable reduction algorithm for behavioral model extraction," in *Proc. IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, 2005, pp. 813–820.
- [115] N. Wang, V. Balakrishnan, and C.-K. Koh, "Passivity-preserving model reduction via a computationally efficient projection-and-balance scheme," in *Proc. IEEE/ACM Design Automation Conference (DAC)*, 2004, pp. 369–374.
- [116] J. Phillips and L. Silveira, "Poor Man's TBR: A simple model reduction scheme," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, pp. 43–55, 2005.
- [117] B. Yan, S. X.-D. Tan, P. Liu, and B. McGaughy, "SBPOR: second-order balanced truncation for passive model order reduction of RLC circuits," in *Proc. IEEE/ACM Design Automation Conference (DAC)*, June 2007, pp. 158–161.
- [118] D. Li, S. X.-D. Tan, and B. McGaughy, "ETBR: Extended truncated balanced realization method for on-chip power grid network analysis," in *Proc. Design, Automation and Test in Europe (DATE)*, 2008, pp. 432–437.
- [119] B. Yan, S. X.-D. Tan, and B. McGaughy, "Second-order balanced truncation for passive-model order reduction of RLCK circuits," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 55, no. 9, pp. 942–946, 2008.
- [120] A.J. Laub, M.T. Heath, C.C. Paige, and R.C. Ward, "Computation of system balancing transformations and other applications of simultaneous diagonalization algorithms," *IEEE Transactions Automatic Control*, vol. 32, pp. 115–122, 1987.
- [121] M.G. Safonov and R.Y. Chiang, "A Schur method for balanced truncation model reduction," *IEEE Transactions on Automatic Control*, vol. 34, pp. 729–733, 1989.

- [122] B. Kagstrom and P.V. Dooren, "A generalized state-space approach for the additive decomposition of a transfer matrix," *Journal of Linear Algebra and Applications*, 1992.
- [123] T. Stykel, "Grammian-based model order reduction for descriptor systems," *Mathematics of Control Signals Systems*, vol. 16, pp. 297–319, 2004.
- [124] K. Willcox and J. Peraire, "Balanced model reduction via the proper orthogonal decomposition," *AIAA Journal*, vol. 40, no. 11, pp. 2323–2330, 2002.
- [125] P. Elias and N. van der Meijs, "Including higher-order moments of RC interconnections in layout-to-circuit extraction," in *Proc. European Design and Test Conf. (DATE)*, 1996, pp. 362–366.
- [126] B.N. Sheehan, "TICER: Realizable reduction of extracted RC circuits," in *Proc. IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, 1999, pp. 200–203.
- [127] C.S. Amin, M.H. Chowdhury, and Y.I. Ismail, "Realizable RLCK circuit crunching," in *Proc. IEEE/ACM Design Automation Conference (DAC)*, 2003, pp. 226–231.
- [128] Z. Qin and C. Cheng, "Realizable parasitic reduction using generalized Y-Delta transformation," in *Proc. IEEE/ACM Design Automation Conference (DAC)*, 2003, pp. 220–225.
- [129] B.N. Sheehan, "Branch merge reduction of RLCM networks," in *Proc. IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, 2003, pp. 658–664.
- [130] S. X.-D. Tan, Z. Qi, and H. Li, "Hierarchical modeling and simulation of large analog circuits," in *Proc. European Design and Test Conference (DATE)*, 2004, pp. 740–741.
- [131] R. Rutenbar, "Next-generation design and EDA challenges," in *Proc. Asia South Pacific Design Automation Conference (ASPDAC)*, January 2007. Keynote speech.
- [132] S. Nassif, "Model to hardware correlation for nm-scale technologies," in *Proc. IEEE International Workshop on Behavioral Modeling and Simulation (BMAS)*, Sept 2007. Keynote speech.
- [133] Y. Liu, L.T. Pileggi, and A.J. Strojwas, "Model order-reduction of RC(L) interconnect including variational analysis," in *Proc. IEEE/ACM Design Automation Conference (DAC)*, 1999, pp. 201–206.
- [134] K. Agarwal, D. Sylvester, D. Blaauw, F. Liu, S. Nassif, and S. Vrudhula, "Variational delay metrics for interconnect timing analysis," in *Proc. IEEE/ACM Design Automation Conference (DAC)*, 2004, pp. 381–384.
- [135] L. Daniel, O.C. Siong, L.S. Chay, K.H. Lee, and J. White, "Multi-parameter moment-matching model-reduction approach for generating geometrically parameterized interconnect performance models," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 23, pp. 678–693, May 2004.
- [136] X. Li, P. Li, and L. Pileggi, "Parameterized interconnect order reduction with explicit-and-implicit multi-parameter moment matching for inter/intra-die variations," in *Proc. IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, 2005, pp. 806–812.
- [137] J.D. Ma and R.A. Rutenbar, "Fast interval-valued statistical interconnect modeling and reduction," in *Proc. International Symposium on Physical Design (ISPD)*, 2005, pp. 159–166.
- [138] J.D. Ma and R.A. Rutenbar, "Fast interval-valued statistical modeling of interconnect and effective capacitance," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, vol. 25, pp. 710–724, April 2006.
- [139] J. Phillips, "Variational interconnect analysis via PMTBR," in *Proc. IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, 2004, pp. 872–879.
- [140] R.E. Moore, *Interval analysis*. Prentice-Hall, 1966.
- [141] S. Vrudhula, J.M. Wang, and P. Ghanta, "Hermite polynomial based interconnect analysis in the presence of process variations," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, pp. 2001–2011, 2006.
- [142] P. Ghanta and S. Vrudhula, "Variational interconnect delay metrics for statistical timing analysis," in *Proc. International Symposium on Quality Electronic Design (ISQED)*, 2006, pp. 19–24.
- [143] N. Mi, J. Fan, and S. X.-D. Tan, "Statistical analysis of power grid networks considering lognormal leakage current variations with spatial correlation," in *Proc. IEEE International Conference on Computer Design (ICCD)*, 2006, pp. 56–62.
- [144] P. Wambacq, G. Gielen, and W. Sansen, "Symbolic simulation of harmonic distortion in analog integrated circuits with weak nonlinearities," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, May 1990, pp. 536–539.
- [145] P. Wambacq and W. Sansen, *Distortion analysis of analog integrated circuits*. Kluwer Academic Publishers, 1998.
- [146] A. Manthe, L. Zhao, C.-J. R. Shi, and K. Mayaram, "Symbolic analysis of nonlinear analog circuits," in *Proc. European Design and Test Conference (DATE)*, 2003, pp. 1108–1109.

CHAPTER 2**Modified Nodal Analysis****Dalibor Bielek^{1,*} and Viera Biolkova²**

¹Dept. of EE/Microelectronics, University of Defense/Brno University of Technology, Brno, Czech Republic and ²Dept. of Radio Electronics, Brno University of Technology, Brno, Czech Republic

Abstract: This chapter gives a description of the Modified Nodal Analysis (MNA) method. Analysis procedures of circuits containing classical and modern circuit elements are described. These methods are explained clearly in a number of examples with a view to the matrix form which is appropriate for computer implementation.

Keywords: Nodal analysis, modified nodal analysis, modeling, admittance matrix, stamp, dead row method, V/I method, symbolic analysis, MATLAB, transistor, OpAmp, OTA, CFA, CCII.

1. Introduction

The up-to-date programs for the analysis and simulation of electronic circuits work, with only minor exceptions, on the basis of Modified Nodal Analysis (MNA) method [1]. The MNA is a universal tool for the analysis of circuits with general nonlinear elements and with current and voltage sources, with the possibility of computing both nodal voltages and currents flowing through selected branches. In view of the focus of this book on symbolic analysis, the following text is targeted at the analysis of linearized circuits.

Since it is advisable to know the principle of the classical Nodal Analysis (NA) method prior to studying the MNA, this Chapter will first summarize the NA, and then its three modifications will be explained. The NA is sufficient to cope with circuits employing arbitrary two-terminal devices with well-defined conductances or admittances, arbitrary multi-terminal devices having the so-called conductance or admittance matrices (such as small-signal linearized transistor models described by y -parameters, voltage-controlled current sources, *etc.*), and conventional current sources. The remaining circuits with elements lacking the conductance or admittance description, such as Operational Amplifiers (OpAmps), should be analyzed *via* the MNA rather than the NA approach.

The first MNA modification, the so-called method of element stamps [1], is highly universal: it can be applied to circuits without any limitation as to the types of circuit element, providing values of nodal voltages as well as currents through arbitrary branches. Its certain drawback consists in rather a large number of equations which must be constructed and solved subsequently. This factor can be limiting for the symbolic analysis of circuit functions based on the determinant expansion, and also for the hand-and-paper computations. Understanding the idea of this method is useful for a better insight into the types of equations the simulation programs based on numerical algorithms work with [2-5]. For the hand-and-paper computations, this type of MNA is used only rarely, for instance when none of the versions described below is allowed due to improper type of circuit elements. A typical example is a circuit containing current conveyors. However, when the user has the possibility of

*Address correspondence to Dalibor Bielek: Dept. of EE/Microelectronics, University of Defense/Brno University of Technology, Brno, Czech Republic; E-mail: dalibor.bielek@unob.cz

utilizing some shareware for symbolic analysis, for example SAPWIN [6] or SNAP [7], they will usually prefer it to the tedious computation.

For the sake of making the symbolic analysis as well as hand-and-paper computations more effective, it is useful to reduce the number of circuit equations *via* a proper selection of the unknown variables which need to be obtained by the analysis. This concept, specified by the abbreviation CMNA (Compacted MNA) [8], is used, for example, in the ISAAC program [9]. The degree of reduction can be different. Two approaches will be described in this Chapter. The first one is the so-called method of the dead row [10, 11], representing a balanced trade-off between the number of circuit equations and the variety of circuit variables being computed. This method is suitable preferably for the analysis of circuits employing ideal voltage amplifiers, including OpAmps. Another modification, the so-called V/I method [5,10,11], provides a minimum number of equations, which can significantly increase the computational speed. It is particularly useful for circuits employing ideal OpAmps.

In this Chapter, the analysis of the behavioral models of circuits employing various active elements (voltage- and current-feedback OpAmps, ideal voltage amplifiers, OTAs, current conveyors) is discussed. The MNA is explained *via* more examples in a simple manner, utilizing equations organized in a matrix form which is suitable for direct implementation in programs for circuit analysis. The methodology described can also be extended to cover analyses of circuits employing behavioral models of other active elements not mentioned here [12].

It is necessary to include here the following note, which concerns all the methods described below. The character of circuit quantities that figure in the equations depends on the type of the circuit being analyzed, and also on the state in which the analyzed circuit should occur. However, the structure of the equations will not depend on the above facts. For example, when solving linear resistive circuits without any accumulative elements, then voltages and currents can be considered as general waveforms, and the circuit elements are described by conductances. For the analysis of linear inertial circuits in the harmonic steady state, the admittance description of each element has been considered, with voltages and currents represented by complex phasors. The most general analysis uses the operator-based circuit model, where the circuit quantities are the Laplace transforms of their time-domain waveforms. The small-signal analysis of nonlinear circuits is characterized by AC components of the waveforms around the DC operating point, also in one of the above forms.

For simplicity, the circuit quantities will be labeled by capital letters throughout the following text, as in the case of DC values, but with the knowledge of the above. When an accumulative element appears in the circuit schematics, we label it by its operator admittance or impedance, considering the circuit quantities to be the operator transforms of their waveforms.

2. Conventional Nodal Analysis (NA)

2.1. The principle of the method

This method is not directly applicable to those circuits which contain sources with known voltages and unknown currents. If possible, these sources must be converted to equivalent current sources prior to writing the circuit equations.

In many cases, we analyze a circuit which does not contain any model of the source, but some driving signal must be considered in order to compute a transfer function,

for example the voltage gain, the input impedance, or another circuit function which is a ratio of two circuit quantities. Then, with regard to the equivalent actions of the voltage and current sources, we can drive the circuit by the current source, which belongs to the “permitted” elements of the NA, even if the voltage source is used in the reality.

The Nodal Analysis is based on the following procedure:

1. One of the circuit nodes is set out as the so-called **reference node** or **common datum node**. The number 0 or the symbol of the ground in computer simulation program is assigned to it. This is a reference node for defining the voltages of all the remaining nodes. These voltages are called **nodal voltages**, forming the set of unknown variables of the method, and the corresponding nodes are called **independent nodes**. For the sake of the simplicity of the algorithm for constructing the equations, it is recommended to direct all the nodal voltages towards the reference node.

Note that the nodal voltages are the only unknown variables even if different circuit variables are the ultimate aim of the analysis. Every voltage and current in the circuit can be derived as a linear combination of nodal voltages. Whereas the simulation program always computes all the unknowns “simultaneously”, even if it is not necessary from the user’s point of view, only the required quantities can be computed by the hand-and-paper method.

2. Kirchhoff’s Current Law (KCL) is applied to each node other than the reference node as follows:

the summation of currents entering the node from external current sources = the summation of currents leaving the node through the circuit branches.

3. The above equations are solved, *i.e.* the nodal voltages are computed, and then the required quantities are calculated from them.

However, it is important that the right-side currents from Item 2 should be derived, with the aid of Ohm’s law, as products of branch admittances/conductances and voltages, and the branch voltages as linear combinations of nodal voltages. As a result, only admittances/conductances and nodal voltages appear on the right-hand sides of the equations. The number of unknowns – nodal voltages – is the same as the number of equations, and it is equal to the number of nodes minus one (the reference node is not taken into account).

2.2. Illustrative example

The NA method will be explained on the example of the circuit from **Fig. 1(a)**. The quantity to be analyzed is current I_{x2} .

Let us first number the nodes. We select the reference node and assign the number 0 to it. Note that the reference node can be chosen arbitrarily. It is mostly selected such that the computed voltage is directly equal to one of the nodal voltages. Also note that the node, at which resistor R_3 is connected with the current source, is actually a reference node, and thus its number is 0. This fact is distinctly marked in the schematics in **Figs. 1(a)** and **(b)**.

After that, we point out the nodal voltages V_1 and V_2 in the schematics, see **Fig. 1(a)**. They form a set of two unknowns, and two equations must be constructed for their computation, the equations of KCL for nodes 1 and 2, to be concrete. Since we are

calculating the current I_{x2} , only the nodal voltage V_2 needs to be computed because the current I_{R3} can directly result from V_2 , and then I_{x2} can be found as a difference of I and I_{R3} .

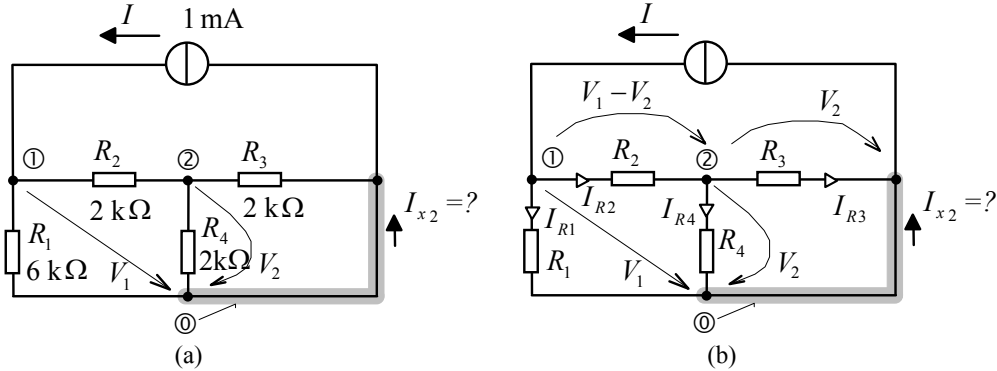


Figure 1: Finding I_{x2} via the Nodal Analysis (NA) method.

According to **Fig. 1 (b)**, the equations of KCL for nodes 1 and 2 are as follows:

$$\textcircled{1}: I = I_{R1} + I_{R2} \quad (1)$$

$$\textcircled{2}: 0 = -I_{R2} + I_{R3} + I_{R4} \quad (2)$$

Note that the branch currents can be directed arbitrarily. When a chosen direction does not correspond to the reality, the computed value will finally differ by the sign. Indeed, it is useful to estimate the real direction of each current prior to constructing the circuit equations.

We derive the right-side branch currents *via* the branch conductances (we use “ G ” symbols with the appropriate suffixes) and the branch voltages which depend on nodal voltages (see **Fig. 1(b)**):

$$\textcircled{1}: I = G_1 V_1 + G_2 (V_1 - V_2) \quad (3)$$

$$\textcircled{2}: 0 = -G_2 (V_1 - V_2) + G_3 V_2 + G_4 V_2 \quad (4)$$

Equations can be arranged in the final form:

$$\textcircled{1}: I = (G_1 + G_2) V_1 - G_2 V_2 \quad (5)$$

$$\textcircled{2}: 0 = -G_2 V_1 + (G_2 + G_3 + G_4) V_2 \quad (6)$$

Substituting the conductances in [mS], the left-side currents will appear in [mA]:

$$\textcircled{1}: 1 = \frac{2}{3} V_1 - 0.5 V_2 \quad (7)$$

$$\textcircled{2}: 0 = -0.5 V_1 + 1.5 V_2 \quad (8)$$

The above equations have the roots:

$$[V_1 \ V_2] = [2 \ 2/3] \text{ V} \quad (9)$$

It follows from **Fig. 1(b)** that when $V_2 = 2/3$ V, then $I_{R3} = 1/3$ mA and the current I_{x2} must be:

$$I_{x2} = I - I_{R3} = \left(1 - \frac{1}{3}\right) \text{mA} = \frac{2}{3} \text{mA} \quad (10)$$

2.3. Rules for algorithmic constructing of the NA equations

Let us generalize the observations from the above example which are important for algorithmic circuit analysis *via* the NA method.

Equation (1) can be expressed in the matrix form:

$$\begin{array}{l} \textcircled{1} \\ \textcircled{2} \end{array} \begin{array}{|c|} \hline I \\ \hline \end{array} = \begin{array}{|cc|} \hline & V_1 & V_2 \\ \hline G_1+G_2 & -G_2 & \\ \hline -G_2 & G_2+G_3+G_4 & \\ \hline \end{array} \begin{array}{|c|} \hline V_1 \\ \hline V_2 \\ \hline \end{array} \quad (11)$$

The right-side square matrix is known as the conductance (or admittance) matrix. Note several rules which should be followed when constructing the matrix equation:

1. Zeros do not need to be recorded. Empty cells are “normal” and their content is automatically regarded as “0”.
2. It is useful to note at the headings of the columns of the conductance matrix the unknowns by which the matrix elements in the given columns are multiplied according to the rule about the matrix-vector multiplication.
3. It is useful to note to the left of the vector of the exciting currents the numbers of nodes, corresponding to the KCL equations.

Comparing the matrix equation with the original circuit schematics in **Fig. 1**, described by this equation, yields important rules. Utilizing them, the set of NA equations can be written directly from the schematics, *i.e.* without any side computations.

Rule for constructing the left-side vector of exciting currents:

- The i -th row contains an algebraic summation of currents flowing into i -th node from external current sources.

Rule for constructing the square conductance (admittance) matrix:

- The “ i,i ” diagonal element contains a summation of all conductances (or admittances) which are connected to node i .
- The “ i,j ” ($i \neq j$) element contains a negatively signed summation of all conductances (or admittances) which are connected directly between nodes i and j .

A note should be added to the latter rule. The fundamental linear two-terminal devices of the R , L , and C types, connected between nodes i and j , are reciprocal in the sense that they behave identically both in “node $i \rightarrow$ node j ” and “node $j \rightarrow$ node i ” directions, or, in other words, their admittances are identical in both cases. That is why the admittance matrices of circuits built up from such elements show a symmetry, *i.e.* the “ i,j ” and “ j,i ” elements are identical. This fact can speed-up the algorithmic construction of the NA equations. However, this feature does not hold when some asymmetric element, for instance diode, appears in the circuit.

The above rules will be demonstrated on the example of rather a complicated circuit in **Fig. 2**. It is a passive 7th-order low-pass ladder filter with a cutoff frequency of 1 kHz.

Four independent nodes are denoted in the schematics. The corresponding nodal voltages are V_1 to V_4 . Applying the above rules yields the set of NA equations in the self-explanatory **Fig. 2**:

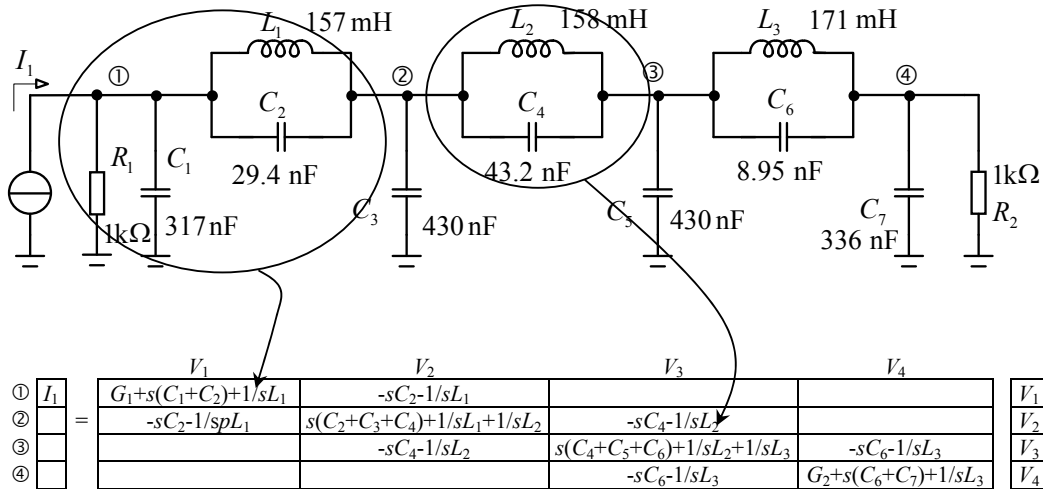


Figure 2: Demonstration of the algorithmic composition of NA equations of ladder filter.

2.4. General structure of the conductance (admittance) matrix

Let us return to the circuit in **Fig. 1 (a)**. **Fig. 3** demonstrates that a given resistive network can be fragmented into individual elements, and that the conductance matrix can be considered a summation of individual conductance matrices of these elements. To put it simply, the conductance matrix of a complicated circuit can be built-up step by step from the matrices of its elements. For example, we will be confronted with the general matrix of linearized model of the Bipolar Junction Transistor (BJT) in section 2.5. After mastering the principles of its stamping into the matrix of the circuit, we will be able to analyze arbitrary linearized circuits containing BJTs.

Take notice of the sub-matrix in **Fig. 3**, which corresponds to the floating resistor R_2 . If we sum all its elements within an arbitrary row or column, we get zero. This is a property of the conductance (admittance) matrix of an arbitrary circuit if the reference node is placed out of this circuit. Such a matrix is then called **complete conductance (admittance) matrix**. If the reference node is additionally declared to be one of the internal nodes of the circuit, say node k , a corresponding new conductance matrix can be obtained from the complete matrix after omitting its k -th row and k -th column. This procedure can be used, for instance, for mutual conversions of transistor parameters in common emitter, base, and collector configurations.

2.5. Matrix-based linearized model of the BJT

A general view of the BJT as a linearized three-terminal device is given in **Fig. 4**.

If the BJT is connected to the circuit in three nodes B , C , and E (base, collector, and emitter), it can be described by three equations of the NA method. The conductances (admittances) $y_{BB} \dots y_{EE}$ in the corresponding matrix model the small-signal transfer parameters of the BJT.

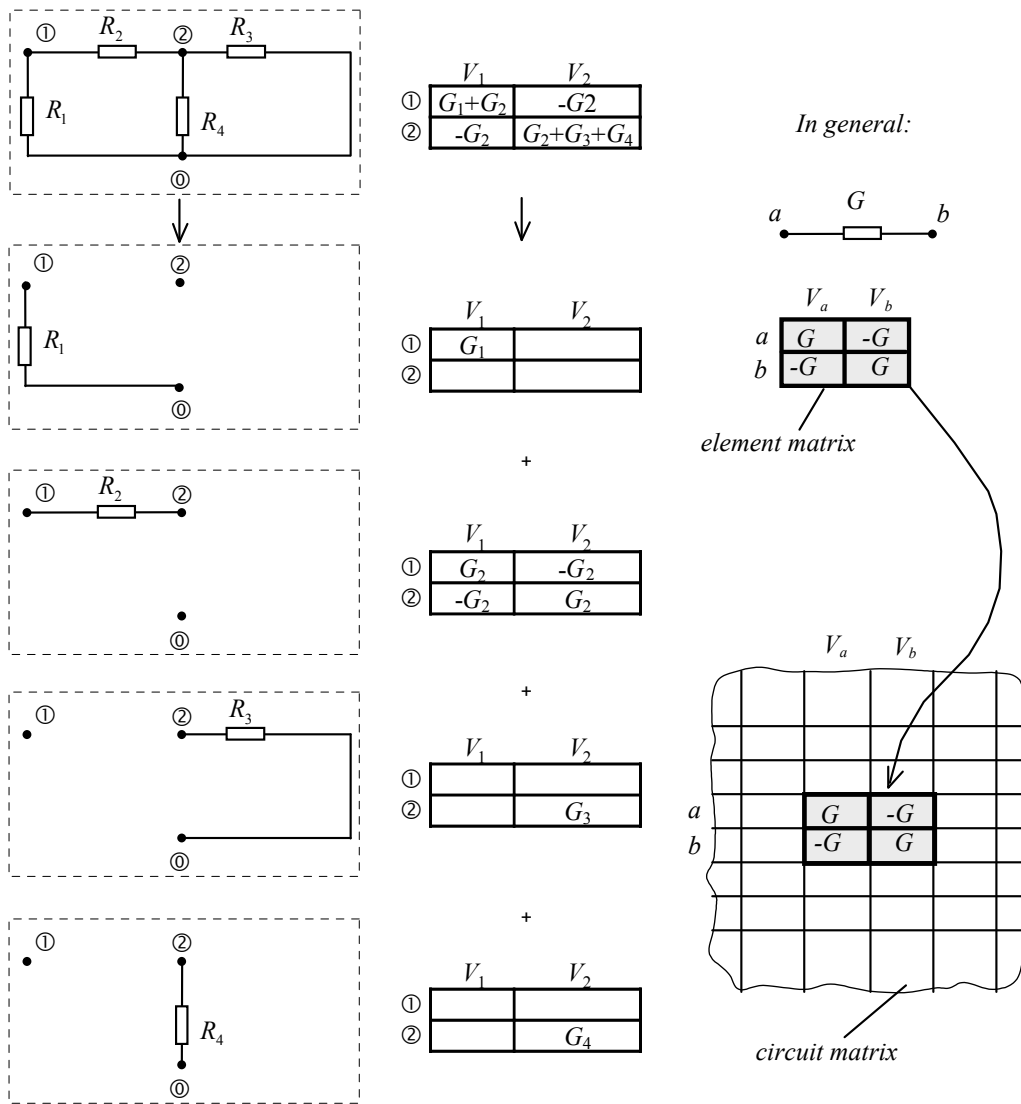


Figure 3: Conductance (admittance) matrix as a summation of matrices of particular elements.

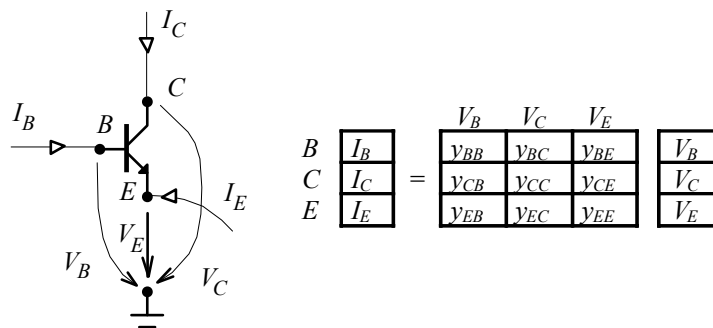


Figure 4: BJT in a general connection, and a set of its linearized equations which correspond to the NA method.

Fig. 5 shows how the set of equations will be modified after connecting the reference node to one of the nodes of the BJT. The common-emitter connection is given in Fig. 5 (a). Since the emitter is grounded, the voltage V_E is zero. Only two equations for nodes B and C are constructed. In the original set of equations, we scratch the equation for node E , and the voltage V_E is not considered any more. The BJT is then

described by a “2x2” admittance matrix, and the matrix elements mean the y -parameters of the BJT in common-emitter connection (suffix e ; base as the input terminal is represented by suffix 1, collector – the output terminal, by suffix 2). These four y -parameters can be obtained either by measurement or by transformation from the known h -parameters.

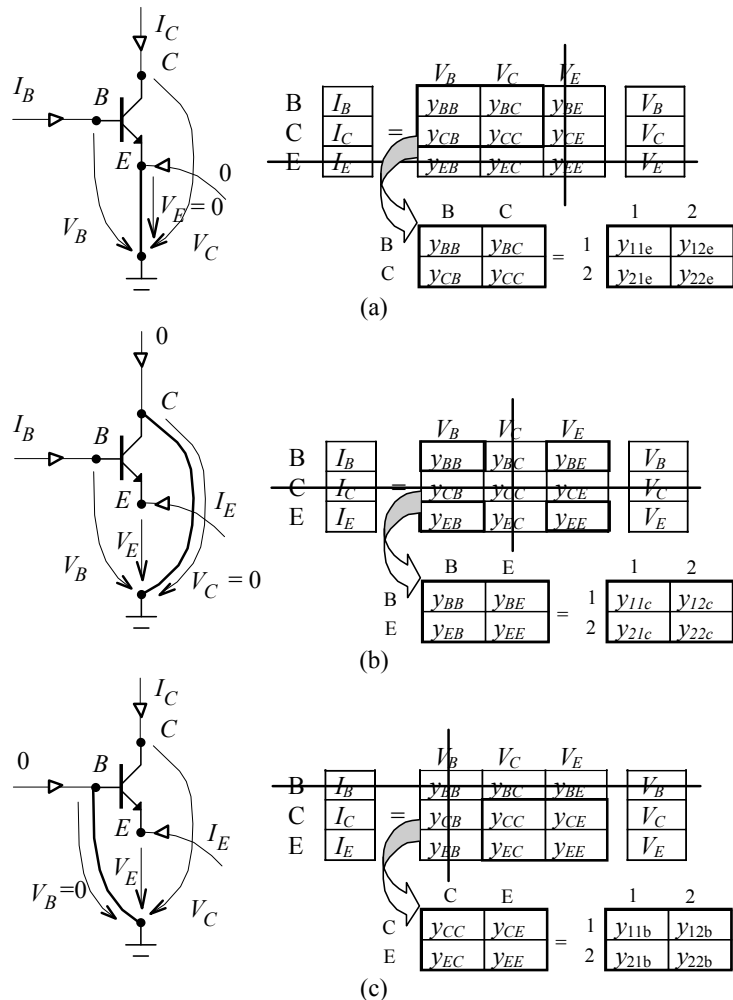


Figure 5: Matrix descriptions of the BJT in common emitter (a), collector (b), and base (c) connections.

The “3x3” admittance matrix is the complete admittance matrix of the transistor, thus the summation of the elements in an arbitrary row or column is zero. Knowing the four parameters y_{BB} , y_{BC} , y_{CB} , and y_{CC} (i.e. four y -parameters of the BJT in common-emitter configuration), the remaining five parameters can be easily evaluated.

Fig. 5(b) and **5(c)** demonstrate the transistor description in common-collector and common-base connections. Only in circuits where all three BJT outlets are floating, will all the 9 transistor parameters figure in the final equations.

2.6. Matrix description versus simplified common modeling of transistor

Let us start from the equations for common-emitter connection in **Fig. 6**. Equations for other variants can be derived from them.

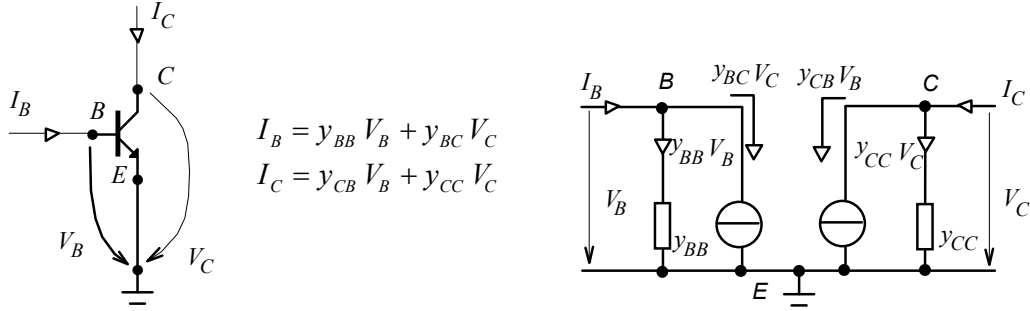


Figure 6: BJT modeling via controlled sources.

These equations can be modeled via controlled sources. Neglecting the parameter y_{BC} , which is justified at low frequencies for most transistors [13], the corresponding controlled source disappears from the equivalent schematics. Then we get a simplified BJT model, which is routinely used for simple computations. The parameter y_{CB} means the BJT transconductance $g_m = \Delta I_C / \Delta U_{BE}$. Matrix description from section 2.5 is more general and, for complex admittances, it also models the BJT behavior for high frequencies. The simplified description is its special case.

For “typical“ values of input resistance, output resistance, and transconductance.

$$r_{BE} \approx 5 \text{ k}\Omega, r_{CE} \approx 100 \text{ k}\Omega, g_m \approx 200 \text{ mA/V} \tag{12}$$

the “typical“ values of y -parameters are as follows:

$$y_{BB} \approx 200 \mu\text{S}, y_{CC} \approx 10 \mu\text{S}, y_{BC} \approx 0, y_{CB} \approx 0.2 \text{ S} \tag{13}$$

$$y_{BE} \approx -200 \mu\text{S}, y_{CE} \approx -0.2 \text{ S}, y_{EB} \approx -200.2 \text{ mS}, y_{EC} \approx -10 \mu\text{S}, y_{EE} \approx 200.2 \text{ mS} \tag{14}$$

2.7. Example of modeling linearized BJT circuits via NA

Let us analyze the linearized model of transistor amplifier in Fig. 7. Consider the BJT parameters (13), (14) at the given operating point.

In the first step, the NA matrix equation is written such that the BJT is not considered in the circuit:

$$\begin{array}{c}
 \textcircled{1} \\
 \textcircled{2} \\
 \textcircled{3} \\
 \textcircled{4}
 \end{array}
 \begin{array}{c}
 I_i \\
 \\
 \\
 \\
 \end{array}
 =
 \begin{array}{c}
 V_1 \quad V_2 \quad V_3 \quad V_4 \\
 \begin{array}{|c|c|c|c|}
 \hline
 G_i + sC_1 & -sC_1 & & \\
 \hline
 -sC_1 & G_B + sC_1 & & \\
 \hline
 & & G_C + sC_2 & -sC_2 \\
 \hline
 & & -sC_2 & G_2 + sC_2 \\
 \hline
 \end{array}
 \begin{array}{c}
 V_1 \\
 V_2 \\
 V_3 \\
 V_4
 \end{array}
 \end{array}
 \tag{15}$$

Subsequently, we stamp the admittance matrix of the BJT into the total matrix. The most convenient way how to do it is as follows: we complete the symbols B and C into rows (on the left side) and columns (on the top) such that the numbers of rows and columns will correspond with the numbers of nodes where base and collector are connected (emitter does not appear here because it is connected to reference node 0, which has no representation in the matrix). Afterwards, we inscribe the individual admittances of the BJT in the appropriate cells of the matrix; the admittance suffixes must correspond to the indices of rows and columns. The result is:

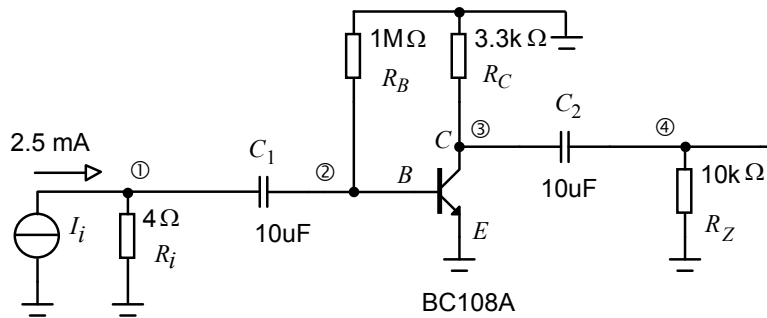


Figure 7: Substitutive small-signal model of transistor amplifier.

		V_1	V_2	B	V_3	C	V_4	
①	I_i	$G_i + sC_1$	$-sC_1$					V_1
②	B	$-sC_1$	$G_B + sC_1 + y_{BB}$	y_{BC}				V_2
③	C		y_{CB}	$G_C + sC_2 + y_{CC}$	$-sC_2$			V_3
④				$-sC_2$	$G_2 + sC_2$			V_4

(16)

The above equation can be used for various computations. After substituting the numerical values of parameters, it becomes the starting point for the calculation of nodal voltages, voltage transfers from the input to all nodes, and impedances, all of them for various frequencies of the input signal, depending on the value of the complex frequency $s = j\omega$. One of the possible methods is described in section 2.9.

The example in **Fig. 8** illustrates a small subcircuit of the linearized model of integrated wideband amplifier RCA 3040. We will show that the above described procedure also works for circuits containing more transistors and even when the transistors are connected in an “atypical” way, for example with variously short-circuited terminals.

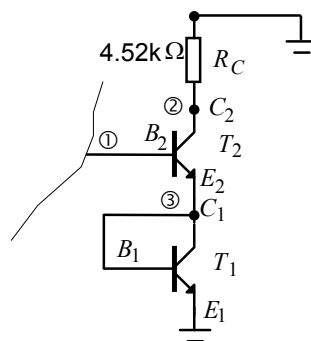


Figure 8: Model of a part of RCA 3040 amplifier.

Let us only describe how the admittance matrix is constructed. The admittance parameters of transistors T_1 and T_2 will be differentiated by superscripts ¹ and ².

First, we construct the matrix of a circuit without BJTs:

	V_1	V_2	V_3	
①				
②		G_C		
③				

(17)

Then the matrix of T_1 will be stamped into it:

	V_1	V_2	V_3	B_1	C_1
①					
②		G_C			
③	B_1	C_1	$y^1_{BB} + y^1_{BC} + y^1_{CB} + y^1_{CC}$		

(18)

Note that all four admittance parameters which follow from the combinations of symbols B_1 and C_1 at the matrix margins are inscribed in the “3, 3” element.

Finally, the matrix of T_2 is stamped into the total matrix of the circuit:

	V_1	B_2	V_2	C_2	V_3	B_1	C_1	E_2
①	B_2	y^2_{BB}	y^2_{BC}		y^2_{BE}			
②	C_2	y^2_{CB}	$G_C + y^2_{CC}$		y^2_{CE}			
③	B_1	C_1	E_2	y^2_{EB}	y^2_{EC}	$y^1_{BB} + y^1_{BC} + y^1_{CB} + y^1_{CC} + y^2_{EE}$		

(19)

2.8. Analysis of circuits containing OTAs

Since the OTA (Operational Transconductance Amplifier), with its schematic symbol shown in **Fig. 9**, behaves as a current source, controlled by the voltage between the input terminals, and because the output current is proportional to this voltage and to the transconductance g_m , the OTA has its admittance matrix (see **Fig. 9**), and thus circuits containing this element can be solved without any problems by the classical NA method.

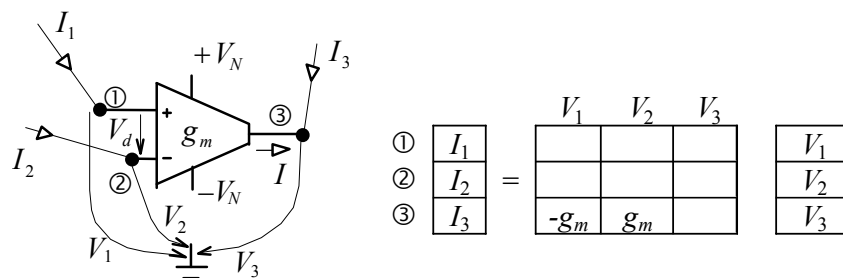


Figure 9: OTA and its matrix description by NA method.

For illustration, we construct equations of the OTA-based filter (the so-called $g_m C$ filter) from **Fig. 10**.

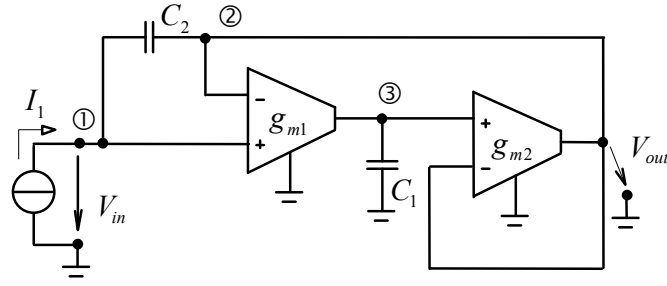


Figure 10: An example of g_mC filter.

The capacitive susceptances and then the submatrices of both OTAs will be written in the “3x3” admittance matrix:

$$\begin{array}{c} \textcircled{1} \\ \textcircled{2} \\ \textcircled{3} \end{array} \begin{array}{c} I_1 \\ \\ \\ \end{array} = \begin{array}{c} V_1 \\ V_2 \\ V_3 \end{array} \begin{array}{ccc} V_1 & V_2 & V_3 \\ \hline sC_2 & -sC_2 & \\ \hline -sC_2 & sC_2 + g_{m2} & -g_{m2} \\ \hline -g_{m1} & g_{m1} & sC_1 \\ \hline \end{array} \begin{array}{c} V_1 \\ V_2 \\ V_3 \end{array} \quad (20)$$

The voltage transfer functions of this filter will be found in the next section, which is devoted to the computation of various circuit functions directly from the admittance matrix.

2.9. The manner of computing the circuit functions from admittance matrix

Constructing the NA equations is the first stage of the analysis. They must be solved subsequently. We will show a practicable method, which is based on the computation of circuit functions using the so-called **signed minors** of the admittance matrix.

The method will be demonstrated on the example of the transistor circuit from **Fig. 11**. Consider the numerical values of y -parameters of both transistors according to (13), (14).

Then admittance matrix (19) of the total circuit will be as follows:

	V_1	V_2	V_3
①	0.2		-0.2
②	200	0.222	-200
③	-200.2	-0.01	400.41

All the admittances are given in millisiemens, and this matrix transforms voltages in volts into currents in milliamperes.

Assume that we find the circuit impedance between node 1 and the ground, and also the voltage gain V_2/V_1 . We connect a current source I_1 to the gate between node 1 and the reference node, we compute the voltage V_1 , caused by this current, and compute the input impedance as a ratio of voltage and current. Then we find V_2 , caused by the input excitation, and determine the gain as a ratio of V_2 and V_1 . The lay-out is exemplified in **Fig. 11**.

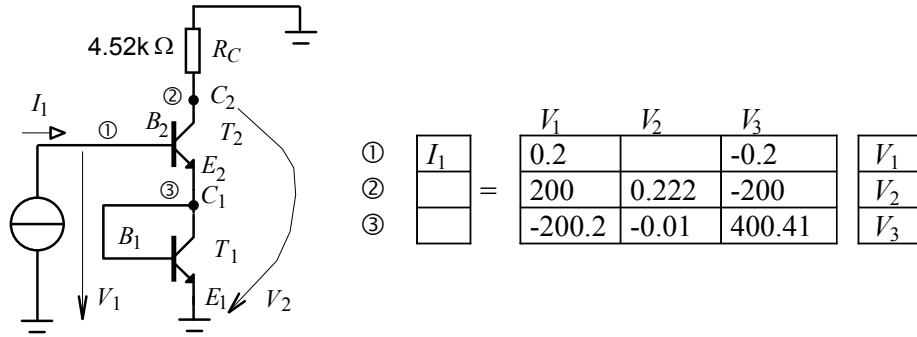


Figure 11: Illustration of the method of computing voltage gain V_2/V_1 .

Note that we do not necessarily need the input voltage source in spite of the voltage gain computation.

We use the **Cramer rule** for computing voltage V_1 from the equation in **Fig. 11**:

The voltage V_k , $k=1,2,3$, is a ratio of two determinants. The admittance matrix determinant Δ is in the denominator. In the numerator, the determinant Δ_k appears, which is computed from a matrix formed from the admittance matrix by replacing the k th column by the vector on the left side of the equation.

The result for V_1 is as follows:

$$V_1 = \frac{\Delta_1}{\Delta} = \frac{\begin{vmatrix} I_1 & 0 & -0.2 \\ 0 & 0.222 & -200 \\ 0 & -0.01 & 400.41 \end{vmatrix}}{\begin{vmatrix} 0.2 & 0 & -0.2 \\ 200 & 0.222 & -200 \\ -200.2 & -0.01 & 400.41 \end{vmatrix}} = \frac{(-1)^{1+1} \begin{vmatrix} 0.222 & -200 \\ -0.01 & 400.41 \end{vmatrix} I_1}{\Delta} = \frac{\Delta_{11}}{\Delta} I_1 \quad (22)$$

The proposition about the expansion of the determinant according to the first column was applied to the numerator.

The symbol Δ_{ij} , specifically $\Delta_{1,1}$ here, represents the so-called **signed minor** of the admittance matrix when eliminating i -th row and j -th column. Numerically, it is equal to the corresponding sub-determinant of the matrix, multiplied by the factor $(-1)^{i+j}$.

Evaluating the determinants yields the result:

$$V_1 \doteq 9.775 I_1 [\text{V,mA}] \Rightarrow Z_1 = \frac{V_1}{I_1} \doteq 9.775 \text{ k}\Omega \quad (23)$$

The impedance (resistance) between nodes 1 and 0 is less than 10 k Ω .

Similarly, we compute the voltage V_2 and the voltage transfer $K = V_2/V_1$.

$$V_2 = \frac{\Delta_2}{\Delta} = \frac{\begin{vmatrix} 0.2 & I_1 & -0.2 \\ 200 & 0 & -200 \\ -200.2 & 0 & 400.41 \end{vmatrix}}{\begin{vmatrix} 0.2 & 0 & -0.2 \\ 200 & 0.222 & -200 \\ -200.2 & -0.01 & 400.41 \end{vmatrix}} = \frac{(-1)^{1+2} \begin{vmatrix} 200 & -200 \\ -200.2 & 400.41 \end{vmatrix} I_1}{\begin{vmatrix} 0.2 & 0 & -0.2 \\ 200 & 0.222 & -200 \\ -200.2 & -0.01 & 400.41 \end{vmatrix}} = \frac{\Delta_{1,2}}{\Delta} I_1 \quad (24)$$

The voltage transfer sought is

$$K = \frac{V_2}{V_1} = \frac{\Delta_{1,2}}{\Delta_{1,1}} = \frac{(-1)^{1+2} \begin{vmatrix} 200 & -200 \\ -200.2 & 400.41 \end{vmatrix}}{(-1)^{1+1} \begin{vmatrix} 0.222 & -200 \\ -0.01 & 400.41 \end{vmatrix}} \doteq -460.8 \quad (25)$$

In the following, let us explain the procedure for computing the output impedance between nodes 2 and 0 for the input gate open.

In this case, the exciting current source I_2 should be connected between nodes 2 and 0. After evaluating the response V_2 , we could determine the impedance Z_2 . The layout is shown in **Fig. 12** together with the modified left side of the equation.

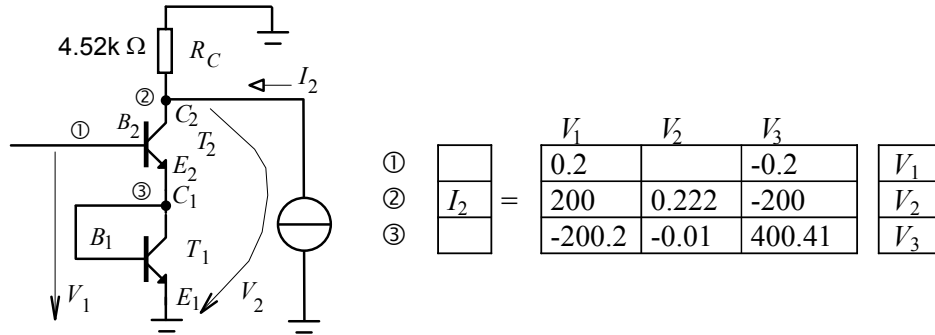


Figure 12: Procedure of the output impedance computation.

The voltage V_2 in volts is now a function of the current I_2 in milliamperes:

$$V_2 = \frac{\begin{vmatrix} 0.2 & -0.2 \\ 200 & I_2 & -200 \\ -200.2 & 0 & 400.41 \end{vmatrix}}{\begin{vmatrix} 0.2 & 0 & -0.2 \\ 200 & 0.222 & -200 \\ -200.2 & -0.01 & 400.41 \end{vmatrix}} = \frac{(-1)^{2+2} \begin{vmatrix} 0.2 & -0.2 \\ -200.2 & 400.41 \end{vmatrix} I_2}{\begin{vmatrix} 0.2 & 0 & -0.2 \\ 200 & 0.222 & -200 \\ -200.2 & -0.01 & 400.41 \end{vmatrix}} = \frac{\Delta_{2,2}}{\Delta} I_2 \doteq 4.505 I_2 \quad (26)$$

and the output impedance (resistance) is:

$$Z_2 = \frac{V_2}{I_2} \doteq 4.505 \text{ k}\Omega \quad (27)$$

On the basis of the above computation, we can state the following rules for calculating the circuit functions from the admittance matrix.

Let us consider a linear circuit with N nodes other than the reference node, and describe it by the “ $N \times N$ ” admittance matrix *via* the NA method. Using the signed minors of this matrix, we can find the following:

The impedance between node k and the reference node:

$$Z_k = \frac{\Delta_{k:k}}{\Delta} \quad (28)$$

The voltage transfer from node i to node o :

$$K = \frac{V_o}{V_i} = \frac{\Delta_{io}}{\Delta_{ii}} \quad (29)$$

The nodal voltage V_k if the circuit is excited by a single current source I_i connected between node i and the reference node:

$$V_k = \frac{\Delta_{ik}}{\Delta} I_i \quad (30)$$

In the above formulae, Δ_{ij} is the signed minor of the admittance matrix when omitting i -th row and j -th column, and Δ is the determinant of the admittance matrix. These formulae are commonly used, enabling direct computations from the admittance matrix without the need to construct the set of equations.

The admittance matrix of OTA-based filter was set up in section 2.8, see also **Fig. 10**. Let us determine the voltage transfers V_2/V_1 and V_3/V_1 .

The first transfer will be:

$$\frac{V_{out}}{V_{in}} = \frac{V_2}{V_1} = \frac{\Delta_{1:2}}{\Delta_{1:1}} = \frac{-(-sC_2sC_1 - g_{m2}g_{m1})}{(sC_2 + g_{m2})sC_1 + g_{m2}g_{m1}} \quad (31)$$

A small re-arrangement yields:

$$\frac{V_2}{V_1} = \frac{s^2 + \omega_0^2}{s^2 + s \frac{\omega_0}{Q} + \omega_0^2} \quad (32)$$

where

$$\omega_0 = \sqrt{\frac{g_{m1}g_{m2}}{C_1C_2}} \quad (33)$$

is the natural frequency of the filter, and

$$Q = \sqrt{\frac{C_2 g_{m1}}{C_1 g_{m2}}} \quad (34)$$

is its quality factor.

It follows from the formulae that, among other things, the filter can be tuned, without violating the quality factor, *via* controlling synchronously the transconductances of both OTAs.

When using the output voltage from node 3, the filter would be of the low-pass type:

$$\frac{V_3}{V_1} = \frac{\Delta_{13}}{\Delta_{11}} = \frac{-sC_2g_{m1} + g_{m1}(sC_2 + g_{m2})}{(sC_2 + g_{m2})sC_1 + g_{m2}g_{m1}} = \frac{\omega_0^2}{s^2 + s\frac{\omega_0}{Q} + \omega_0^2} \quad (35)$$

3. Modified Nodal Analysis (MNA)

An advantage of the NA method is that it can be easily implemented in computer programs for circuit analysis: the algorithm of constructing the set of equations directly from the circuit schematics is very simple. However, this method does not enable analyzing circuits containing voltage sources and other elements that lack the admittance matrix, such as transformers, various operational amplifiers, conveyors, and other modern active components.

That is why the classical NA must undergo some modification which would preserve the advantage of simple algorithm but also enable an analysis of the circuit without the above limitations. Three approaches will be described below: from the most universal but computationally rather expensive, to a modification enabling a very effective analysis of a relatively limited class of circuits.

3.1. The stamp method

Each “problematic” element is described by M additional equations, $M \geq 1$. The number of unknown variables is also increased by M . Some of the equations of KCL are also modified. Then the matrix equation will be of a special structure: The original admittance matrix will be extended with rows and columns, whose elements are of different dimensions than “siemens”. This extension contains the so-called matrix-stamps of additional elements. The resulting matrix is then called the **pseudo-admittance matrix**.

The increased size of the set of equations need not be a problem for computer analysis based on the numerical principle. However, this is not the case of symbolic and hand-and-paper analyses.

Consider a circuit described by equations of classical NA. A general two-terminal device, described by the Thévenin model according to **Fig. 13**, is additionally included between nodes a and b . This inclusion causes changes in the voltage and current relations in the circuit. The current I_x which modifies voltages and currents associated with nodes a and b will flow through this device; the original nodal voltages will be also modified.

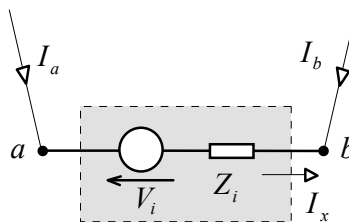


Figure 13: Inclusion of general two-terminal device, described by Thévenin model, into the circuit.

The original equation, describing the current equilibrium at node a , must be completed on the right-hand side by a current I_x , flowing out of the node, and at node b by a current I_x with negative sign because it flows inside the node b . In addition, the nodal voltages V_a and V_b are now tied by the following condition:

$$Z_i I_x + V_b = V_i + V_a \tag{36}$$

or

$$V_i = Z_i I_x + V_b - V_a \tag{37}$$

All the above modifications can be included in a new set of equations of MNA (Modified Nodal Analysis):

$$\begin{array}{c} a \\ b \\ \cdot \\ \cdot \\ V_i \end{array} = \begin{array}{c|c|c|c|c} & V_a & V_b & & I_x \\ \hline I_a & & & \dots & +1 \\ I_b & & & \dots & -1 \\ \cdot & & & \cdot & \\ \cdot & & & \cdot & \\ \hline -1 & 1 & \dots & & Z_i \end{array} \begin{array}{c} V_a \\ V_b \\ \\ \\ I_x \end{array} \tag{38}$$

stamp \rightarrow

The vector of unknown nodal voltages is extended by another quantity, current I_x . The number of equations is also incremented due to the above coupling condition between nodal voltages V_a and V_b . The voltage V_i is included in the left-side vector of exciting quantities. The modification of KCL equations for nodes a and b is accomplished *via* recording the numerals +1 and -1 in the column “ I_x ”.

The above procedure can give guidance how to analyze, *via* the MNA method, circuits containing *e.g.* voltage sources. The impedance Z_i can also be zero, and then the voltage source will be ideal. The case when $V_i = 0$ and simultaneously $Z_i = 0$ models a short circuit between the nodes, and the MNA can compute the current flowing through this connection. This approach can be used, for example, for the analysis of circuits containing current-controlled sources.

In cases when more elements without admittance description appear in the circuit, each of them is modeled by its matrix-stamp, and the size of the pseudo-admittance matrix is increasing.

This method will be characterized in detail *via* several examples.

3.1.1. Passive circuits containing independent current and voltage sources

Let us analyze the circuit from **Fig. 14** *via* the MNA method. In comparison with the circuit from **Fig. 1(a)**, it is extended by voltage source V . What is sought is the current I_x flowing out of the voltage source.

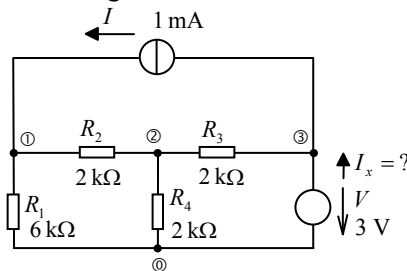


Figure 14: The circuit under analysis.

There are three independent nodes and one reference node in the circuit. We construct the equation of classical NA for each independent node. Then we add the coupling

condition between nodal voltages caused by the voltage source, namely that the nodal voltage V_3 is a source voltage, and we modify the original equations by the influence of current I_x , the equation for node 3 to be concrete. The result is:

$$\begin{array}{l}
 \textcircled{1} \\
 \textcircled{2} \\
 \textcircled{3}
 \end{array}
 \begin{array}{c}
 I \\
 \\
 -I \\
 V
 \end{array}
 =
 \begin{array}{c}
 V_1 \quad V_2 \quad V_3 \quad I_x \\
 \begin{array}{|c|c|c|c|}
 \hline
 G_1+G_2 & -G_2 & & \\
 \hline
 -G_2 & G_2+G_3+G_4 & -G_3 & \\
 \hline
 & -G_3 & G_3 & 1 \\
 \hline
 & & 1 & \\
 \hline
 \end{array}
 \end{array}
 \begin{array}{c}
 V_1 \\
 V_2 \\
 V_3 \\
 I_x
 \end{array}
 \quad (39)$$

We set the numerical values; For simplicity, conductances in [mS] and currents in [mA]:

$$\begin{array}{l}
 \textcircled{1} \\
 \textcircled{2} \\
 \textcircled{3}
 \end{array}
 \begin{array}{c}
 1 \\
 \\
 -1 \\
 3
 \end{array}
 =
 \begin{array}{c}
 V_1 \quad V_2 \quad V_3 \quad I_x \\
 \begin{array}{|c|c|c|c|}
 \hline
 2/3 & -0.5 & & \\
 \hline
 -0.5 & 1.5 & -0.5 & \\
 \hline
 & -0.5 & 0.5 & 1 \\
 \hline
 & & 1 & \\
 \hline
 \end{array}
 \end{array}
 \begin{array}{c}
 V_1 \\
 V_2 \\
 V_3 \\
 I_x
 \end{array}
 \quad (40)$$

The analysis of the above set of equations gives the result $I_x = 1.5$ mA.

3.1.2. Circuits employing ideal voltage-feedback OpAmps

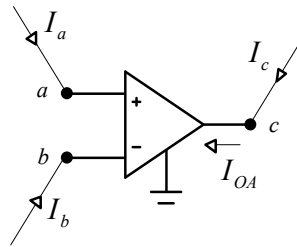


Figure 15: Ideal voltage-feedback OpAmp.

The ideal Voltage-Feedback operational Amplifier (VFA) in **Fig. 15**, included in a circuit and operating in the linear regime, will cause an equality of nodal voltages V_a and V_b , and also a modification of current relations at node c :

$$\begin{array}{l}
 a \\
 b \\
 c \\
 . \\
 . \\
 \hline
 \end{array}
 \begin{array}{c}
 I_a \\
 I_b \\
 I_c \\
 \\
 \\
 \hline
 \end{array}
 =
 \begin{array}{c}
 V_a \quad V_b \quad V_c \quad I_{OA} \\
 \begin{array}{|c|c|c|c|}
 \hline
 & & & \dots & \\
 \hline
 & & & & \\
 \hline
 & & & \dots & 1 \\
 \hline
 & & & & \\
 \hline
 & & & & \\
 \hline
 1 & -1 & & \dots & \\
 \hline
 \end{array}
 \end{array}
 \begin{array}{c}
 V_a \\
 V_b \\
 V_c \\
 \\
 \\
 \hline
 I_{OA}
 \end{array}
 \quad (41)$$

The following equation is written in the additional bottom row:

$$0 = 1.V_a - 1.V_b. \tag{42}$$

Naturally, the analysis result remains unchanged when both sides of this equation are multiplied by an arbitrary nonzero number. That is why, for example $[-1 \ 1]$ or possibly $[15 \ -15]$ may be in the bottom row instead of $[1 \ -1]$. At this moment, we do not consider the impact of such modifications on the numerical precision of computation.

Connecting one of the OpAmp inputs to the reference node results in that the corresponding nodal voltage does not appear in the set of equations, and only a single “1” will figure in the last row instead of the common couple $[1 \ -1]$. Such a case will be demonstrated in the following example.

The numeral “1” in row c and column I_{OA} represents the addition of current I_{OA} to the total current balance of node c .

For the next illustration, let us mention the inverting amplifier containing a T -cell in **Fig. 16**. It is necessary to find the voltage gain V_4/V_1 .

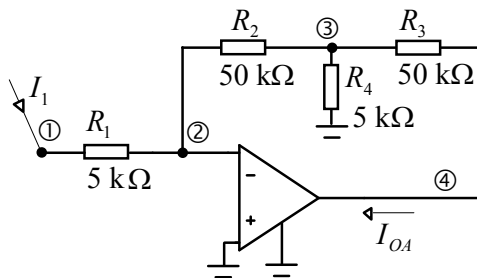


Figure 16: Amplifier employing the T -cell.

Suppose that the amplifier is excited into node 1. For simplicity, we select a current source. In the case of voltage source, one additional equation would be necessary. The MNA equations will be as follows:

		V_1	V_2	V_3	V_4	I_{OA}	
①	I_1	G_1	$-G_1$				V_1
②		$-G_1$	G_1+G_2	$-G_2$			V_2
③			$-G_2$	$G_2+G_3+G_4$	$-G_3$		V_3
④				$-G_3$	G_3	1	V_4
			1				I_{OA}

(43)

The numeral “1” in the additional row represents a simple formula $V_2 = 0$.

This example demonstrates the inefficiency of the given method for hand-and-paper analysis. In spite of the circuit simplicity, we have a set of 5 equations with 5 unknowns. Nevertheless, the additional equation indicates that the unknown variable V_2 can be excluded because it is zero. Basically, we are not interested in the current I_{OA} either, and yet it appears on the vector of unknown variables.

The computation yields:

$$K = \frac{V_4}{V_1} = -\frac{R_2 + R_3 + \frac{R_2 R_3}{R_4}}{R_1} = -120 \tag{44}$$

3.1.3. Circuits employing ideal voltage amplifiers (IVA)

The ideal voltage amplifier in **Fig. 17** has the following features:

- infinite input resistance, which yields zero input currents,
- zero output resistance, thus the IVA output behaves as an ideal voltage source,
- the output voltage is equal to the product of the gain A and the differential input voltage.

The ideal voltage-feedback OpAmp is thus a special case of the IVA for the gain increasing to infinity. The unity-gain voltage buffer is another special case, which can be modeled *via* the IVA on the assumption of grounded inverting input and unity gain A .

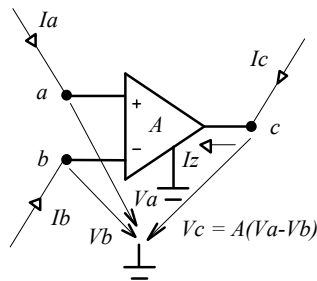


Figure 17: Ideal voltage amplifier.

Stamping the IVA equations onto the set of MNA equations will be similar to the ideal OpAmp. The only difference consists in the matrix-stamp because the voltage coupling condition is now different:

$$0 = V_c - A \cdot V_a + A \cdot V_b \tag{45}$$

The final set of equations for the general circuit containing an IVA has the following structure:

		V_a	V_b	V_c	I_Z	
a	I_a				...	V_a
b	I_b					V_b
c	I_c				...	V_c
.	=					
.						
		$-A$	A	1	...	I_Z

(46)

Let us try to find the voltage gain of the amplifier in **Fig. 18**. The MNA leads to four equations:

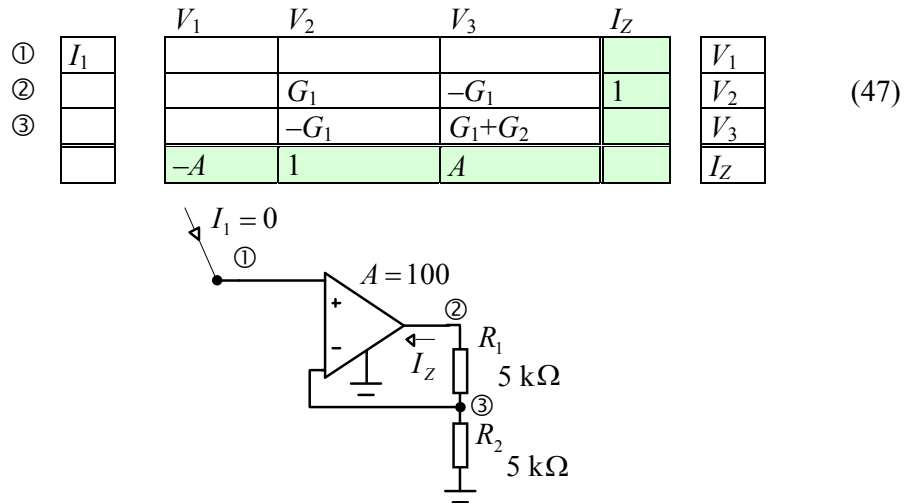


Figure 18: A circuit containing one IVA.

The required gain can be computed *via* signed minors:

$$\frac{V_2}{V_1} = \frac{\Delta_{1,2}}{\Delta_{1,1}} = \frac{\begin{vmatrix} 0 & -G_1 & 1 \\ 0 & G_1+G_2 & 0 \\ -A & A & 0 \end{vmatrix}}{\begin{vmatrix} G_1 & -G_1 & 1 \\ -G_1 & G_1+G_2 & 0 \\ 1 & A & 0 \end{vmatrix}} = \frac{-A(G_1+G_2)}{G_1+G_2-AG_1} \doteq 2.04 \quad (48)$$

Note that the gain would be exactly 2 for ideal OpAmp (if $A \rightarrow \infty$).

3.1.4. Circuits employing current conveyors CCII

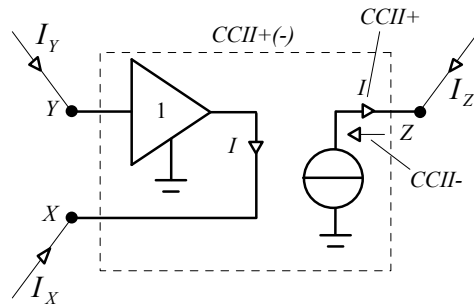


Figure 19: Positive and negative current conveyors CCII.

Fig. 19 shows a model of ideal positive (negative) current conveyor CCII+(-). The current I serves as an additional unknown variable in the set of MNA equations. Inserting the conveyor into the circuit makes the voltages V_X and V_Y equal (unity-gain buffer acting between the X and Y terminals), and the current relations at nodes X and Z will be also modified.

The matrix-stamp of the current conveyor is as follows:

$$\begin{array}{c} X \\ Y \\ Z \\ \cdot \\ \cdot \\ \cdot \end{array} \begin{array}{c} I_X \\ I_Y \\ I_Z \\ \cdot \\ \cdot \\ \cdot \end{array} = \begin{array}{c|ccc|c} & V_X & V_Y & V_Z & I \\ \hline & & & \dots & -1 \\ & & & & \\ & & & \dots & -(+)1 \\ & & & \cdot & \\ & & & \cdot & \\ \hline & 1 & -1 & \dots & \end{array} \begin{array}{c} V_X \\ V_Y \\ V_Z \\ \cdot \\ \cdot \\ I \end{array} \quad (49)$$

The numeral “1” in row “Z” and column “I” has the sign – for the positive and + for the negative current conveyor.

Let us find the current gain I_{out}/I_{in} of the “Sallen-Key” current-mode filter in **Fig. 20**.

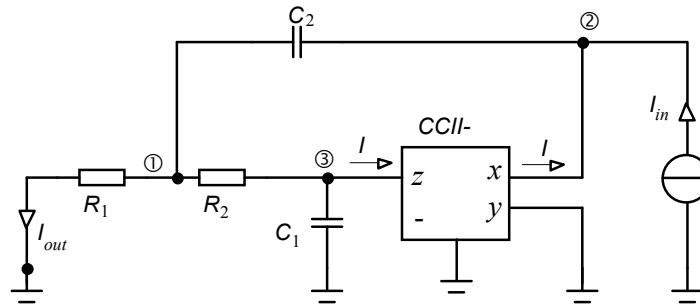


Figure 20: Active Sallen-Key 2nd order filter employing the current conveyor.

We construct the set of MNA equations:

$$\begin{array}{c} \textcircled{1} \\ \textcircled{2} \\ \textcircled{3} \\ \cdot \\ \cdot \end{array} \begin{array}{c} \\ I_{in} \\ \\ \cdot \\ \cdot \end{array} = \begin{array}{c|ccc|c} & V_1 & V_2 & V_3 & I \\ \hline & G_1+G_2+sC_2 & -sC_2 & -G_2 & \\ & -sC_2 & sC_2 & & -1 \\ & -G_2 & & G_2+sC_1 & 1 \\ \hline & & 1 & & \end{array} \begin{array}{c} V_1 \\ V_2 \\ V_3 \\ I \end{array} \quad (50)$$

We compute V_1 and I_{out} :

$$\begin{aligned} V_1 = \frac{\Delta_{2:1}}{\Delta} I_{in}, I_{out} = G_1 V_1 \Rightarrow \frac{I_{out}}{I_{in}} = G_1 \frac{\Delta_{2:1}}{\Delta} = G_1 \frac{\begin{vmatrix} -sC_2 & -G_2 & 0 \\ 0 & G_2 + sC_1 & 1 \\ 1 & 0 & 0 \end{vmatrix}}{\Delta} = \quad (51) \\ = \frac{G_1 G_2}{s^2 C_1 C_2 + s C_1 (G_1 + G_2) + G_1 G_2} = \frac{1}{s^2 R_1 R_2 C_1 C_2 + s C_1 (R_1 + R_2) + 1} \end{aligned}$$

Note that the analyzed circuit is a low-pass 2nd order filter. The dependence of the natural frequency ω_0 and the quality factor Q on the parameters of passive components can be derived from the above formulae.

3.1.5. Circuits employing current-feedback OpAmp (CFA)

Since the Current-Feedback operational Amplifier (CFA) consists of a positive current conveyor CCII+ and a voltage buffer (see Fig. 21), the matrix-stamp of the CFA could be derived from the matrices of the two components. However, we can make use of the fact that the internal current I of the CCII is zero for ideal CFA (the transimpedance acting in parallel to the current source is infinity, thus the current cannot flow through it). As a simplification, we need not consider this unknown variable. However, we must take into account the output current of the terminal buffer instead.

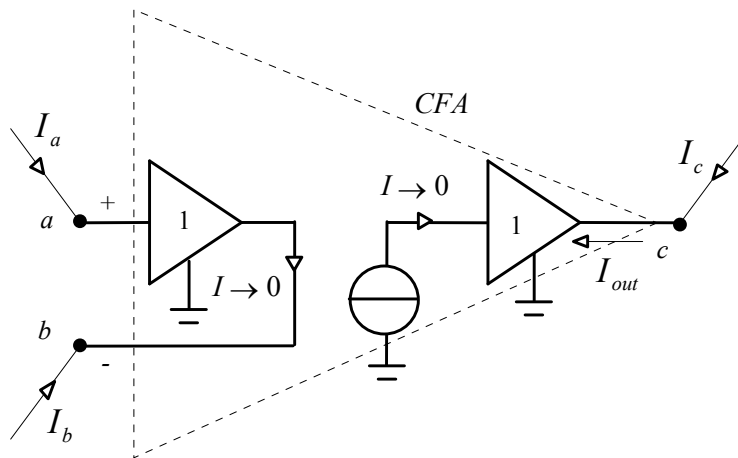


Figure 21: Idealized model of the current-feedback OpAmp.

The MNA equations of the CFA are below:

		V_a	V_b	V_c	I_{out}	
a	I_a					V_a
b	I_b					V_b
c	I_c					V_c
	.					
	.					
		1	-1	...		I_{out}

(52)

For illustration, the inverting amplifier from Fig. 22 will be analyzed.

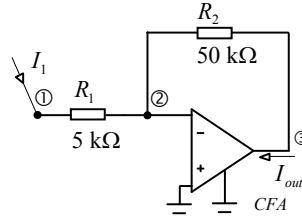


Figure 22: Analyzed circuit employing the CFA.

The MNA equations are now

$$\begin{array}{c} \textcircled{1} \\ \textcircled{2} \\ \textcircled{3} \end{array} \begin{array}{|c|} \hline I_1 \\ \hline \\ \hline \\ \hline \end{array} \begin{array}{|c|c|c|c|} \hline V_1 & V_2 & V_3 & I_{out} \\ \hline G_1 & -G_1 & & \\ \hline -G_1 & G_1+G_2 & -G_2 & \\ \hline & -G_2 & G_2 & 1 \\ \hline & 1 & & \\ \hline \end{array} \begin{array}{|c|} \hline V_1 \\ \hline V_2 \\ \hline V_3 \\ \hline I_{out} \\ \hline \end{array} \quad (53)$$

The voltage gain is the same as for the similar circuit employing the classical voltage-feedback amplifier:

$$\frac{V_3}{V_1} = \frac{\Delta_{1:3}}{\Delta_{1:1}} = \frac{\begin{vmatrix} -G_1 & G_1+G_2 & 0 \\ 0 & -G_2 & 1 \\ 0 & 1 & 0 \end{vmatrix}}{\begin{vmatrix} G_1+G_2 & -G_2 & 0 \\ -G_2 & G_2 & 1 \\ 1 & 0 & 0 \end{vmatrix}} = \frac{G_1}{-G_2} = -\frac{R_2}{R_1} = -10 \quad (54)$$

Attentive reader might notice that the matrix-stamps of voltage- and current-feedback OpAmps are identical. Indeed, the ideal VFA and CFA behave identically in the linear regime because of their identical properties – zero differential voltages, zero input currents, zero output resistances, and infinite voltage gain. That is why the circuit principle is not violated when interchanging these ideal OpAmps in linear applications. However, the impact of real properties can be very different [14].

The stamp method is very general, as is obvious from the above illustrations. It enables the analysis of circuits without any fundamental limitations as to the type of internal elements. In addition, this method can be easily implemented in computer simulation programs.

On the other hand, one should be also aware of several drawbacks. The increasing number of equations is uncomfortable, particularly for symbolic and hand-and-paper analyses. This increase is often “needless”. For example, the matrix-stamp of the OpAmp adds the output current as a new unknown variable even if it may not be the aim of the analysis. One additional equation is included, which states that two voltages at OpAmp inputs are equal. However, both these voltages are included in the vector of unknown quantities and they are computed as independent values. It is obvious that this method can be improved from the point of view of an economical construction of circuit equations.

For the above reasons, other procedures were developed which can be more useful for the hand-and-paper and algorithmic symbolic analyses. Two of them are described in sections 3.2 and 3.3.

3.2. The dead row method

In contrast to the method of stamps, the dead row method preserves the same number of equations as in the classical NA method, which is equal to the number of independent nodes. It is especially useful for the analysis of circuits containing ideal OpAmps and ideal voltage amplifiers. As shown in section 3.4, this method can be also used for the analysis of current conveyor applications.

Let us consider the ideal OpAmp in **Fig. 23**. As shown in section 3.1, all the considerations, associated with this linear model, hold true for the VFA and also CFA types.

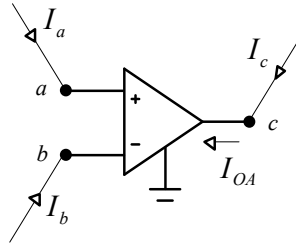


Figure 23: Ideal OpAmp of the VFA or CFA type.

In sections 3.1.2 and 3.1.5, the following matrix-stamp was derived:

$$\begin{array}{c}
 \begin{array}{c} a \\ b \\ c \\ \cdot \\ \cdot \\ \cdot \end{array} \begin{array}{c} I_a \\ I_b \\ I_c \\ \cdot \\ \cdot \\ \cdot \end{array} = \begin{array}{c|c|c|c|c|c} & V_a & V_b & V_c & & I_{OA} \\ \hline & & & & \dots & \\ \hline & & & & & \\ \hline & & & & \dots & 1 \\ \hline & & & & \cdot & \\ \hline & & & & \cdot & \\ \hline & 1 & -1 & & \dots & \end{array} \begin{array}{c} V_a \\ V_b \\ V_c \\ \cdot \\ \cdot \\ I_{OA} \end{array} \quad (55)
 \end{array}$$

Note that there is only one element – “1” – in the column “ I_{OA} “, namely in the row “ c “. It means that this variable is used only in the equation for node c . In other words, the unknown variable I_{OA} can be computed from this equation, and it is not necessary for computation in other equations.

If I_{OA} is not the aim of analysis, we need neither the equation in row “ c “ nor the unknown I_{OA} . To reduce the set of equations, we record in the matrix the additional row of the OpAmp stamp instead of the row “ c “, and the variable I_{OA} will be omitted from the vector of unknown variables. The result is:

$$\begin{array}{c}
 \begin{array}{c} a \\ b \\ c \\ \cdot \\ \cdot \\ \cdot \end{array} \begin{array}{c} I_a \\ I_b \\ 0 \\ \cdot \\ \cdot \\ \cdot \end{array} = \begin{array}{c|c|c|c|c} & V_a & V_b & V_c & \\ \hline & & & & \dots \\ \hline & & & & \\ \hline & 1 & -1 & & \dots \\ \hline & & & & \cdot \\ \hline & & & & \cdot \end{array} \bullet \begin{array}{c} V_a \\ V_b \\ V_c \\ \cdot \\ \cdot \\ \cdot \end{array} \quad (56)
 \end{array}$$

The row “c”, called the **dead row**, is denoted by a special sign, for example by a dark circlet. In contrast to the remaining rows, it is not allowed to write down here any admittances according to the algorithm of classical MNA. The reason is simple – we cannot violate the voltage coupling condition, $0 = 1 \cdot V_a - 1 \cdot V_b$, which is the only one defined in the dead row.

The same principle may be applied to constructing the equations in the case of ideal voltage amplifier, where only the type of coupling condition is different. The same procedure is useful for independent voltage sources. Let us generalize the above method for circuits employing OpAmps, IVA, and ideal voltage sources.

Practical procedure for implementing the dead row method:

- 1) Label the node numbers in the schematics. The reference node will be assigned the number 0.
- 2) Find the number of independent nodes, *i.e.* the number of all nodes minus one. Sketch the frame of the matrix equation, fill in the right-side vector of nodal voltages, the left-side vector of exciting currents, and headings of rows and columns.
- 3) Identify the index of the node to which the output of OpAmp, or IVA, or grounded ideal voltage source is connected. The corresponding row will be denoted the dead row. In the case of more such elements, each of them will be represented by its dead row. Exclude the case of interconnecting the outputs of ideal voltage amplifiers and sources.
- 4) The voltage coupling condition, which corresponds to the amplifier or voltage source, is written in the dead row.
- 5) Fill in the remaining elements of the matrix according to the classical algorithm known from the NA method. However, avoid all the elements from the dead rows.

The above procedure is illustrated on the example of positive impedance converter from **Fig. 24**, where a formula for the input impedance is required.

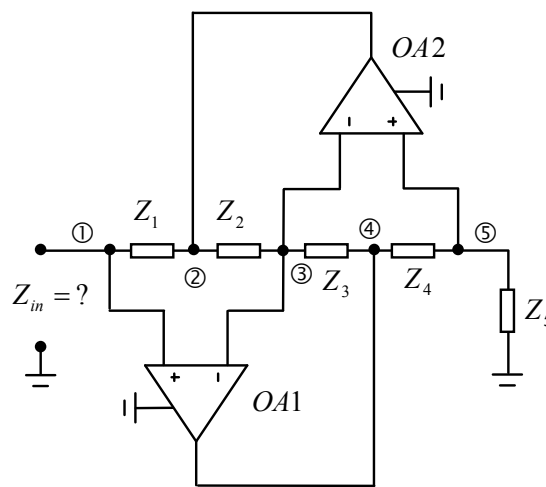


Figure 24: Positive impedance converter.

	V_1	V_2	V_3	V_4	V_5	
①	Y_1	$-Y_1$				
②			1		-1	● OA2
③		$-Y_2$	Y_2+Y_3	$-Y_3$		
④	1		-1			● OA1
⑤				$-Y_4$	Y_4+Y_5	

(57)

Dead row No. 2 (4) belongs to OpAmp OA2 (OA1).

The input impedance can be computed from the pseudo-admittance matrix:

$$Z_{in} = \frac{\Delta_{11}}{\Delta} = \frac{\begin{vmatrix} 0 & 1 & 0 & -1 \\ -Y_2 & Y_2+Y_3 & -Y_3 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -Y_4 & Y_4+Y_5 \end{vmatrix}}{\Delta} = \frac{Z_1 Z_3}{Z_2 Z_4} Z_5 \quad (58)$$

Note that the dead row method can lead to an effective analysis of circuits containing voltage amplifiers. The following section describes another, extra space-saving method, which is particularly useful for circuits employing ideal operational amplifiers.

3.3. The V/I method

This method is based on the so-called **method of voltage and current graphs** [5], or method of voltage and current coefficients [11]. One of its versions, optimized for practical computations, is described below. Every ideal OpAmp in the circuit will decrease the number of equations by one. This feature can significantly reduce the computation time.

The principle is outlined in **Fig. 25**. Instead of all nodal voltages, only some of them will be included in the set of unknown variables. Concretely, only one of the input voltages of the OpAmp is considered (because they are equal) or none of them, when one of the inputs is grounded (such a voltage is then zero). The reduction of unknowns must be accompanied by the reduction of equations: the KCL equations are omitted for nodes which correspond to the OpAmp outputs. The explanation is simple. Such equations are omitted and they are not replaced by any dead rows, since the dead rows would contain equations where “the differential input voltage of the OpAmp is zero“. These equations are now directly substituted by reducing the identical voltages and by a procedure which can be summarized as follows:

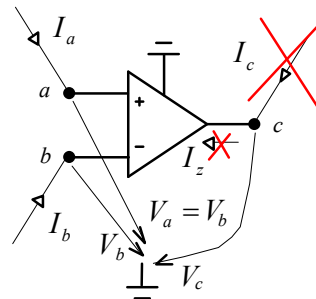


Figure 25: Illustration of the V/I method.

- 1) Label the reference node and also the remaining nodes in the circuit schematics. Find the total number of equations, which equals the number of independent nodes (*i.e.* without counting the reference node) minus the number of OpAmps in the circuit.
- 2) Sketch the frame of the matrix equation or, alternatively, only the frame of the admittance matrix. The rows will be labeled in ascending order by the node numbers, but omitting the nodes which correspond to the OpAmp outputs. The symbols of nodal voltages will be inscribed in the heads of columns. If some nodal voltages are equal, for example V_2 and V_3 , due to connecting floating OpAmp inputs to them, write $V_2 = V_3$ in the head of the column. If some nodal voltage is zero as a consequence of the OpAmp “virtual ground”, this voltage does not appear in the equations at all.
- 3) The matrix elements are filled in *via* the classical algorithm known from the NA approach, but all the combinations of the indices in rows and columns must now be taken into account.

Consider once again the positive impedance converter from **Fig. 24**. This circuit has 5 independent nodes, but it contains 2 OpAmps. That is why we construct only $5 - 2 = 3$ equations for nodes 1, 3, and 5 (the OpAmp outputs are connected to nodes 2 and 4). The unknown quantities will also be 3: $V_1 = V_3 = V_5$, V_2 , V_4 . The above procedure yields a compact matrix

$$\begin{array}{l}
 \textcircled{1} \\
 \textcircled{3} \\
 \textcircled{5}
 \end{array}
 \begin{array}{|c|c|c|}
 \hline
 V_1 = V_3 = V_5 & V_2 & V_4 \\
 \hline
 Y_1 & -Y_1 & \\
 \hline
 Y_2 + Y_3 & -Y_2 & -Y_3 \\
 \hline
 Y_4 + Y_5 & & -Y_4 \\
 \hline
 \end{array}
 \quad (59)$$

The input impedance is now computed more economically than by the dead row method:

$$Z_{in} = \frac{\Delta_{1,1}}{\Delta} = \frac{\begin{vmatrix} -Y_2 & -Y_3 \\ 0 & -Y_4 \end{vmatrix}}{\begin{vmatrix} Y_1 & -Y_1 & 0 \\ Y_2 + Y_3 & -Y_2 & -Y_3 \\ Y_4 + Y_5 & 0 & -Y_4 \end{vmatrix}} = \frac{Z_1 Z_3}{Z_2 Z_4} Z_5 \quad (60)$$

The power of the V/I method is also illustrated in the following example of inverting amplifier employing a *T*-cell from **Fig. 16**. The unknown quantities are now V_1 , V_3 , and V_4 (V_2 is omitted because it is zero). Only KCL equations for nodes 1, 2, and 3 are written (the OpAmp output is connected to node 4). The result is in the form

$$\begin{array}{l}
 \textcircled{1} \\
 \textcircled{2} \\
 \textcircled{3}
 \end{array}
 \begin{array}{|c|c|c|}
 \hline
 V_1 & V_3 & V_4 \\
 \hline
 G_1 & & \\
 \hline
 -G_1 & -G_2 & \\
 \hline
 & G_2 + G_3 + G_4 & -G_3 \\
 \hline
 \end{array}
 \quad (61)$$

We determine the voltage transfer from node 1 to node 4. A formal problem can appear when working with signed minors: The numbers of omitted rows and columns correspond with the node numbers, but not with the sequence numbers of rows and columns. We should be careful both when selecting the omitted rows and columns and when determining the signs of signed minors.

$$\frac{V_4}{V_1} = \frac{\Delta_{1,4}}{\Delta_{1,1}} = \frac{\begin{vmatrix} -G_1 & -G_2 \\ 0 & G_2 + G_3 + G_4 \end{vmatrix}}{\begin{vmatrix} -G_2 & 0 \\ G_2 + G_3 + G_4 & -G_3 \end{vmatrix}} = \frac{-G_1(G_2 + G_3 + G_4)}{G_2G_3} = -\frac{R_2 + R_3 + \frac{R_2R_3}{R_4}}{R_1} \quad (62)$$

3.4. Analysis of circuits employing current conveyors in greater detail

Second generation current conveyors (CCII) with their wide application potential in voltage-, current-, and mixed- mode analog signal processing belong to useful active elements. In the analysis of their application circuits, the choice between special programs for symbolic analysis and stamped-based MNA is not comfortable. The following text is devoted to those interested in a fast analysis of CCII-based circuits. Several hitherto unpublished procedures, based on the philosophy of the dead row and the V/I method will be explained here.

Recall the matrix-stamp, which was derived in section 3.1.4 for positive and negative current conveyors CCII:

		V_X	V_Y	V_Z	I	
X	I_X			...	-1	V_X
Y	I_Y					V_Y
Z	I_Z			...	-(+1)	V_Z
.	.			.		
.	.					
		1	-1	...		I

(63)

Recall that the sign in row “Z” and column “I” is minus for the positive and plus for the negative conveyor.

The dead row method consisted in omitting the KCL equation for the output node of the amplifier in which an auxiliary current appeared, and this equation was replaced by a voltage coupling condition. However, this causes both a fundamental and a technical problem in the case of circuits containing current conveyors.

The fundamental problem consists in the necessity of removing the output current of the conveyor from the set of unknown quantities. However, this current, particularly in current-mode applications, is a frequently computed quantity. On the other hand, such a current can be computed additionally from voltage relations.

A technical problem also appears because current I now occurs in two equations, not only in the KCL equation for output node Z , but also in the equation for node X . This problem can be overcome by subtraction or addition of the equations for nodes Z and X in order to cancel the numeral “-1” in row X and column I .

For the CCII+ conveyor, the equation for node Z will be subtracted from the equation for node X . For CCII-, these equations will be summed. The following intermediate result is obtained:

$$\begin{array}{l} X, -(+)Z \\ Y \\ Z \end{array} \begin{array}{c} I_X - (+)I_Z \\ I_Y \\ I_Z \\ \cdot \\ \cdot \end{array} = \begin{array}{c} V_X \quad V_Y \quad V_Z \quad \dots \quad I \\ \begin{array}{c|c|c|c|c} \hline & & & \dots & \\ \hline & & & & \\ \hline & & & & \\ \hline & & & & \\ \hline & & & & \\ \hline 1 & -1 & & \dots & \\ \hline \end{array} \end{array} \begin{array}{c} V_X \\ V_Y \\ V_Z \\ \cdot \\ \cdot \\ I \end{array} \quad (64)$$

The equation for node Z can be omitted together with the unknown I, and the auxiliary equation can be placed instead. The given row will have the status of dead row:

$$\begin{array}{l} X, -(+)Z \\ Y \\ Z \end{array} \begin{array}{c} I_X - (+)I_Z \\ I_Y \\ \cdot \\ \cdot \end{array} = \begin{array}{c} V_X \quad V_Y \quad V_Z \\ \begin{array}{c|c|c|c} \hline & & & \dots \\ \hline & & & \\ \hline 1 & -1 & & \dots \\ \hline & & & \cdot \\ \hline & & & \cdot \end{array} \end{array} \bullet \begin{array}{c} V_X \\ V_Y \\ V_Z \end{array} \quad (65)$$

The above procedure can be generalized as follows:

- 1) The current conveyor does not modify the number of equations and unknowns of the classical NA method, which is equal to the number of independent nodes. That is why we sketch the frame of matrix equation, fill in the vector of unknown nodal voltages and the vector of exciting currents, and inscribe the node indices and symbols of nodal voltages in the heads of rows and columns.
- 2) Let us denote the row whose number is equal to the number of the Z-output node of CCII, as the dead row. We inscribe here the coupling condition that voltages at inputs X and Y are identical.
- 3) In the left margin of row X we type the number of the Z-output node of CCII, and complete the information about the sign (depending on whether the conveyor is positive or negative). If this node is excited by an external current source, we add this current – with the corresponding sign – to the vector of exciting currents.
- 4) The remaining part of the admittance matrix, except for the dead row, will be filled in according to the classical NA algorithm. However, we must consider all the combinations of the indices in row X, and if some index is accompanied by the “minus” sign, the corresponding admittance must be written with opposite sign.

The procedure will be explained *via* the already analyzed current-mode filter from **Fig. 20**. Its current gain $I_{out}/I_{in} = G_1 \cdot U_1/I_{in}$ was computed in section 3.1.4. The final set of equations is in the following form:

$$\begin{array}{l} \textcircled{1} \\ \textcircled{2}, \textcircled{3} \\ \textcircled{3} \end{array} \begin{array}{c} \\ I_{in} \\ \cdot \end{array} = \begin{array}{c} V_1 \quad V_2 \quad V_3 \\ \begin{array}{c|c|c} \hline G_1+G_2+sC_2 & -sC_2 & -G_2 \\ \hline -sC_2-G_2 & sC_2 & G_2+sC_1 \\ \hline & 1 & \end{array} \end{array} \bullet \begin{array}{c} V_1 \\ V_2 \\ V_3 \end{array} \quad (66)$$

The X-terminal of the negative CCII is connected to node 2 whereas the Z-terminal to node 3. That is why we add the number 3 on the left side of row 2. Row 3 is the dead

row, with inscribed equation “voltage of node 2 is zero“. The remaining elements of the matrix will be filled in according to the classical algorithm. The elements in row 2 are added according to the following template: The element in column “ V_1 “ is equal to “– admittance between nodes 2-1 – admittance between nodes 3-1“. The element in column “ V_2 “ is equal to “+ admittance connected to node 2 – admittance between nodes 3-2“, *etc.*

The required current gain will be computed from the set of equations:

$$V_1 = \frac{\Delta_{2,1}}{\Delta} I_{in}, I_{out} = G_1 V_1 \Rightarrow \frac{I_{out}}{I_{in}} = G_1 \frac{\Delta_{2,1}}{\Delta} = G_1 \frac{- \begin{vmatrix} -sC_2 & -G_2 \\ 1 & 0 \end{vmatrix}}{\Delta} = \quad (67)$$

$$= \frac{1}{s^2 R_1 R_2 C_1 C_2 + s C_1 (R_1 + R_2) + 1}.$$

The above example indicates that CCII-based circuits could be also analyzed *via* the V/I method. For example, it is needless to consider the unknown variable V_2 because we know that it is zero. We save one unknown and one equation. Each current conveyor decreases the number of equations by one.

Practical procedure for implementing the V/I method for circuits employing CCII:

- 1) Label the reference node and enumerate the remaining nodes. Compute the total number of equations, which is equal to the number of independent (numbered) nodes minus the number of current conveyors.
- 2) Sketch the frame of the matrix equation or, alternatively, only the frame of the admittance matrix. The rows will be labeled in ascending order by the node numbers, but omitting the nodes which correspond to the Z -outputs of CCII. The symbols of nodal voltages will be inscribed in the heads of columns. If some nodal voltages are equal, for example V_2 and V_3 , due to connecting floating CCII inputs X and Y to them, we write $V_2 = V_3$ in the head of the column. If some nodal voltage is zero as a consequence of connecting the Y terminal to the reference node, this voltage does not appear in the equations.
- 3) In the left margin of row X , type the number of the Z -output node of CCII, and complete the information about the sign (depending on whether the conveyor is positive or negative). If this node is excited by an external current source, add this current – with the corresponding sign – to the vector of exciting currents.
- 4) The elements of the admittance matrix will be filled in according to the classical NA algorithm. However, one must consider all the combinations of the indices in the rows and columns, and if some index is accompanied by the “minus” sign, the corresponding admittance must be written with opposite sign.

The equations of the filter from **Fig. 20** would now be as follows:

$$\begin{array}{l} \textcircled{1} \\ \textcircled{2}, \textcircled{3} \end{array} \begin{array}{c} \boxed{} \\ \boxed{I_{in}} \end{array} = \begin{array}{c|c} V_1 & V_3 \\ \hline G_1+G_2+sC_2 & -G_2 \\ -sC_2-G_2 & G_2+sC_1 \end{array} \begin{array}{c} \boxed{V_1} \\ \boxed{V_3} \end{array} \quad (68)$$

The current gain computation is now very simple and thus we do not mention it here.

Let us analyze a more complicated circuit, the CCII-based gyrator in **Fig. 26**. It is necessary to check that the input impedance Z_1 is inversely proportional to the loading impedance Z .

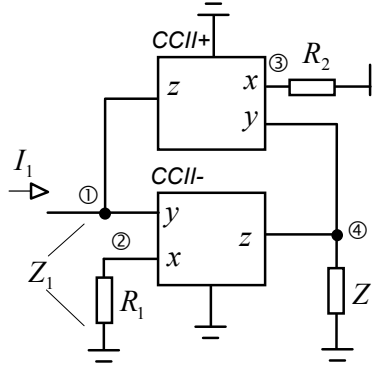


Figure 26: Gyrator employing two current conveyors.

The admittance matrix constructed in accordance with the above procedure is in the form:

$$\begin{array}{l} \textcircled{1}, -\textcircled{3} \\ \textcircled{2}, \textcircled{4} \end{array} \begin{array}{c|c} V_1 = V_2 & V_3 = V_4 \\ \hline -G_2 \\ G_1 & Y \end{array} \quad (69)$$

The input impedance is:

$$Z_1 = \frac{U_1}{I_1} = \frac{\Delta_{11}}{\Delta} = \frac{Y}{G_1 G_2} = \frac{R_1 R_2}{Z} \quad (70)$$

Note that this circuit contains two different current conveyors and that it has four nodes plus the reference node. The stamp method would lead to a set of six equations whereas the dead row method leads to four equations.

3.5. Analysis of circuits with magnetic couplings

Magnetic couplings are frequently modeled *via* circuits with mutual inductances or ideal transformers. In both cases, the symbolic or hand-and-paper analysis by the NA method is rather problematic. The circuits employing mutual inductances are more advantageously analyzed by the method of mesh currents. Simple circuits containing ideal transformers can be solved intuitively. The modified nodal analysis is a universal method, particularly for automated computer-aided numerical analysis of such circuits.

Note from the theory that when the circuit contains magnetic couplings, it can be generally divided into m galvanically separated parts, $m \geq 1$. Then the modified theorem of the number of independent nodes in the circuits holds:

$$\text{Number of independent nodes} = \text{number of nodes} - m. \tag{71}$$

Until this time, only compact circuits have been considered, with $m = 1$. This theorem is important for the practical implementation of the NA method, because it gives information about the number of equations and the number of unknown nodal voltages, and also about the possible way of selecting these voltages. The possibility (not necessarily utilized) of assigning more reference nodes appears: each separated part can have its own reference node.

3.5.1. Circuits with coupled inductors

The well-known model of two inductors with mutual inductance M is illustrated in **Fig. 27**. The reference node is considered in its most general position, *i.e.* outside the circuit element. The dots in the schematics denote the so-called beginnings of windings. They define the direction of voltage induced on the given inductor, which is caused by a current flowing through the other inductor.

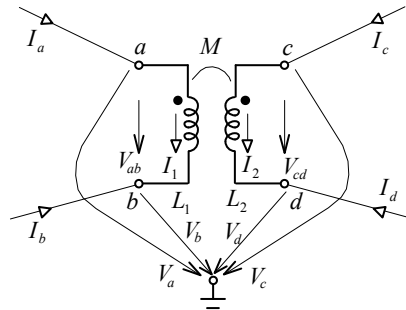


Figure 27: Model of the circuit with mutual inductance M .

Equations of the operator-type can be written for this circuit. After the substitution $s = j\omega$, they can be also used for computations in the harmonic steady state:

$$V_{ab} = sL_1 I_1 + sM I_2 \tag{72}$$

$$V_{cd} = sL_2 I_2 + sM I_1 \tag{73}$$

The mutual inductance M is coupled with the self-inductances L_1 and L_2 by the relation:

$$M = k\sqrt{L_1 L_2} \tag{74}$$

where k is the coupling coefficient, *i.e.* the number within the interval from 0 to 1.

With regard to **Fig. 27**, equations (72) and (73) can be rewritten in the matrix form

		V_a	V_b	V_c	V_d	I_1	I_2		
I_a	a					1			V_a
I_b	b					-1			V_b
I_c	c						1		V_c
I_d	d						-1		V_d
		-1	1			sL_1	sM		I_1
				-1	1	sM	sL_2		I_2

$\tag{75}$

Every magnetic coupling causes an increase in the number of equations by two, and also the same increase in the number of unknown quantities, namely the currents through the inductors.

Consider the Campbell filter in **Fig. 28(a)**. It is necessary to find the amplitude and the initial phase of the current through R_2 at a frequency of 5 kHz. The initial phase of voltage V is zero.

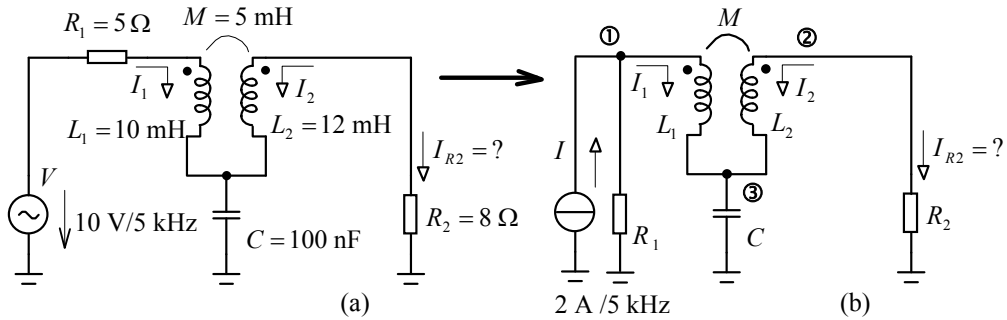


Figure 28: (a) Campbell filter, (b) voltage-to-current source transformation as a step towards an economical utilization of MNA.

The voltage source in **Fig. 28(a)** is transformed into the equivalent current source in **Fig. 28(b)**, which reduces the number of independent nodes to three. The MNA equations are as follows:

		V_1	V_2	V_3	I_1	I_2						
I	=	①	G_1			1		V_1				
		②		G_2			1		V_2			
		③			$j\omega C$	-1	-1			V_3		
			-1		1	$j\omega L_1$	$j\omega M$				I_1	
				-1	1	$j\omega M$	$j\omega L_2$					I_2

(76)

The current I_{R2} can be computed from the determinants:

$$I_{R2} = -I_2 = -\frac{\Delta_{1:5}}{\Delta} I \tag{77}$$

A demonstration of computing the determinants *via* MATLAB is shown below, with the final result:

$$I_{R2} = 61.4 \angle -90.6^\circ \text{ mA} \tag{78}$$

Numerical analysis of (76) in MATLAB:

```

R1=5;R2=8;L1=10e-3;L2=12e-3;
M=5e-3;C=100e-9;I=2;           defining the circuit parameters
f=5000;om=2*pi*f;
BC=j*om*C;XL1=j*om*L1;XL2=j*om*L2;XM=j*om*M; computing the reactances
G1=1/R1;G2=1/R2;
MATRIX=[G1 0 0 1 0;           defining the pseudo-admittance matrix
 0 G2 0 0 1;
 0 0 BC -1 -1;
 -1 0 1 XL1 XM;
 0 -1 1 XM XL2];
deter=det(MATRIX);           computing the determinant Δ
deter1=det(MATRIX(2:5,1:4)); computing the signed minor Δ1:5
IR2= -I*deter1/deter;       computing the current IR2
abs(IR2)                    computing the magnitude of current IR2
phase(IR2)*180/pi           computing the initial phase of current IR2

```

Note that the original circuit in **Fig. 28(a)** can be easily analyzed by hand *via* the mesh currents method, which would lead to only 2 equations. The MNA works with 5 equations. It is obvious that the MNA cannot be recommended for a hand-and-paper analysis of such circuits.

Let us add that there exists a possibility of specially arranging the MNA equations in order to solve circuits with mutual inductances more economically. Equations (72), (73) can be inverted, *i.e.* deriving currents I_1 and I_2 from voltages, and transferring these currents to the left side. Then we get admittance equations, containing the so-called **inverse inductances** described by rather complicated formulae. The number of these equations will be the same as the number of independent nodes after including them in the MNA. Then the Campbell filter from **Fig. 28(b)** would be described by three equations. However, it is still more than with the method of mesh currents.

3.5.2. Circuits containing ideal transformers

Ideal transformer (*IT*) is a substantial idealization of the circuit with mutual inductors from **Fig. 27**. No stray magnetic flux (*i.e.* the magnetic coupling coefficient $k = 1$), no losses in magnetic circuit and windings, and a core with infinite magnetic conductivity are assumed. The last assumption implies that all inductances (L_1 , L_2 , and M) are infinite. The corresponding inductive reactances/susceptances are thus infinite/zero, which rules out direct utilization of classical methods of analysis – the NA method, and also the mesh current approach.

As follows from **Fig. 29**, the *IT* is defined by the numbers of turns N_1 and N_2 of both coils, or by the **turns ratio**:

$$n = \frac{N_2}{N_1}. \quad (79)$$

The following elementary equations hold true for *IT*:

Voltage transformation:

$$N_1 V_{cd} = N_2 V_{ab} \quad (80)$$

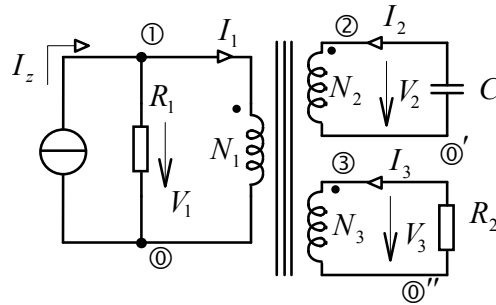


Figure 30: Circuit with ideal transformer. $R_1 = 10 \Omega$, $R_2 = 8 \Omega$, $C = 20 \mu\text{F}$, $I_z = 20 \text{ A}/50 \text{ Hz}$, $N_1 = 1420$ turns, $N_2 = 150$ turns, $N_3 = 80$ turns.

The set of equations of the circuit in **Fig. 30** is in the form.

		V_1	V_2	V_3	I_1	I_2	I_3	
I_z	①	G_1			1			V_1
	②		sC			1		V_2
	③			G_2			1	V_3
		$-N_2$	N_1					I_1
		$-N_3$		N_1				I_2
					N_1	N_2	N_3	I_3

(88)

The Cramer rule yields all the unknown variables. The procedure is evident from the list of the MATLAB M-file below. The magnitudes of voltages and currents are as follows:

- $V_1 = 199.2 \text{ V}$
- $V_2 = 21.0 \text{ V}$
- $V_3 = 11.2 \text{ V}$
- $I_1 = 80.3 \text{ mA}$
- $I_2 = 132.2 \text{ mA}$
- $I_3 = 1.4 \text{ A}$.

Numerical analysis of (88) in MATLAB:

```

R1=10;R2=8;C=20e-6;N1=1420;N2=150;N3=80;    defining the circuit parameters
f=50;I=20;
om=2*pi*f;
matrix=[1/R1 0 0 1 0 0;
        0 j*om*C 0 0 1 0;
        0 0 1/R2 0 0 1;
        -N2 N1 0 0 0 0;
        -N3 0 N1 0 0 0;
        0 0 0 N1 N2 N3];
deter=det(matrix);
deter11=det(matrix(2:6,2:6));
deter12=det(matrix(2:6,[1 3:6]));
deter13=det(matrix(2:6,[1 2 4:6]));
    defining the pseudo-admittance matrix
    computing the determinant Δ
    computing the signed minor Δ1:1
    computing the signed minor Δ1:2
    computing the signed minor Δ1:3
    
```

deter14=det(matrix(2:6,[1:3 5:6]));	<i>computing the signed minor $\Delta_{1:4}$</i>
deter15=det(matrix(2:6,[1:4 6]));	<i>computing the signed minor $\Delta_{1:5}$</i>
deter16=det(matrix(2:6,[1:5]));	<i>computing the signed minor $\Delta_{1:6}$</i>
U1=abs(I*deter11/deter)	<i>computing the magnitude V_1</i>
U2=abs(I*deter12/deter)	<i>computing the magnitude V_2</i>
U3=abs(I*deter13/deter)	<i>computing the magnitude V_3</i>
I1=abs(I*deter14/deter)	<i>computing the magnitude I_1</i>
I2=abs(I*deter15/deter)	<i>computing the magnitude I_2</i>
I3=abs(I*deter16/deter)	<i>computing the magnitude I_3</i>

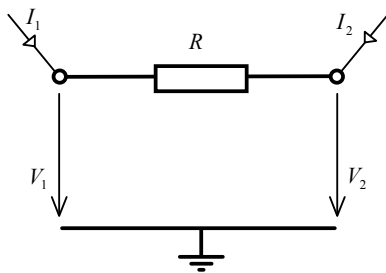
4. Demonstration of creating models of circuit components based on MNA for computer-aided symbolic analysis

A concrete implementation of the stamp method of the MNA in the SNAP shareware [15] for symbolic analysis is described in this section.

SNAP provides the symbolic analysis of linear circuits without any limitations as to the type of circuit elements. Simultaneously, it generates all the circuit functions which assume the computing of arbitrary voltages and currents in the circuit. That is why the mathematical models of circuit components are based on the general method of stamps, described in section 3.1. The models are saved in an auxiliary text file snap.cdl.

For example, the resistor model is as follows (see **Fig. 31**):

[R]	model name
2	number of nodes
1 1	size of the parameter matrix
0	number of additional circuit variables
MAT	starting point of the circuit matrix definition
1 1 2 2 1 1 1 0 -1	definition of the circuit matrix



$$\begin{bmatrix} I_1 \\ I_2 \end{bmatrix} = \begin{bmatrix} 1/R & -1/R \\ -1/R & 1/R \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \end{bmatrix}$$

Figure 31: Matrix-stamp of floating resistor in SNAP.

The model definition has the following structure:

[*model name*]

number of nodes connecting the element with its surroundings

size of the matrix r (number of rows, number of columns) where all element parameters are given; for the resistor, only one parameter is considered (resistance R).

number of additional circuit variables that are necessary for component modeling using the MNA; for the resistor, no additional variable is needed.

MAT – special keyword, which defines the beginning of the definition of the matrix elements of MNA.

Definition of the circuit matrix:

The matrix elements are defined using a special convention – the so-called "atom definition". The general structure is as follows:

$a\ b\ c\ d\ k\ i\ j\ n\ m$

4 variables

$k\ i\ j\ n\ m$

define a matrix element – so-called atom, in accordance with the equation

$$\text{atom} = (k * r[i,j] * s^{(n)})^m$$

where,

$r[i,j]$.. element parameter located in the i -th row and j -th column of parameter matrix r

s .. Laplace operator

n, m .. integers.

For example, the inductor susceptance is $1/(sL)$. Consider a single parameter of the inductor – inductance, which is saved in the simple matrix of parameters $r(1 \times 1)$. Then the susceptance is defined in the following row:

$k\ i\ j\ n\ m\ \dots\dots\ 1\ 1\ 1\ 1\ -1$

4 variables

$a\ b\ c\ d$

define the "atom" location in the circuit matrix according to the following rules:

"Atom" appears at the intersections of

a -th row and b -th column, with the sign +

c -th row and d -th column, with the sign +

a -th row and d -th column, with the sign –

c -th row and b -th column, with the sign –

on the assumption that neither of the pair of row and column numbers is zero.

In the opposite case, the "atom" is not copied into the matrix.

In the next example, the model of ideal operational amplifier (OpAmp) is shown in **Fig. 32**. Together with nodal voltages, one auxiliary circuit variable is used – OpAmp output current I . The ideal OpAmp does not require any circuit parameters, the parameter matrix r is thus empty.

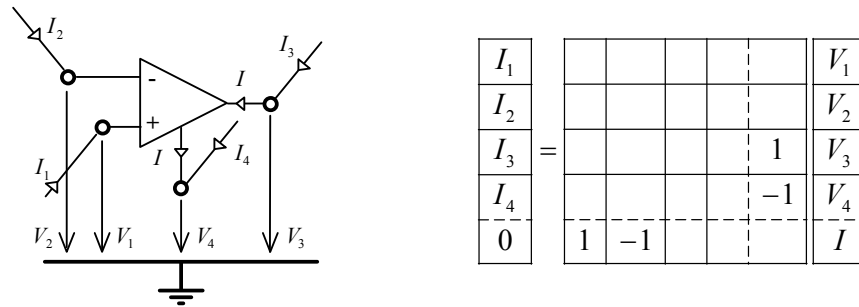


Figure 32: Matrix-stamp of ideal OpAmp in SNAP.

```
[OPA]
4
0 0
1
MAT
-1 1 0 2 1 0 0 0 1
3 -1 4 0 1 0 0 0 1
```

The sign – preceding some of the row or column numbers in the MAT command represents the symbol of an additional row or column of the circuit matrix. For example, –1 refers to the additional row (or column) No.1.

We recommend downloading the installation files of SNAP program from [15], looking into the file snap.cdl, and studying the technique of implementing the models of various circuit components. All the cases represent practical applications of the stamp MNA method. Through mastering this technique, you will be able to script your own models of arbitrary components.

5. Conclusion

The hand-and-paper analysis is particularly useful for checking computations in circuits with simple modeling of individual ideal components. Computer-aided analysis is recommended in all the other cases.

The decision whether intuitive or algorithmic procedures should be used for the hand-and paper analysis is, more or less, subjective. Somebody likes the intuitive approach, which is tailored for people who are used to solving problems "in their own original style", creatively utilizing the basic laws and principles of Electrical Engineering. Others prefer algorithmic methods, which always lead to a solution but usually at the cost of uncomfortable routine computation.

As the third alternative, any analysis task can be resolved *via* a proper computer program.

When we come to the conclusion that the intuitive procedures are beyond our abilities, or if we do not prefer them for any other reason, then we can consider the computer analysis or the hand-and-paper algorithmic analysis, described in this Chapter. Computer analysis becomes necessary in the case of large circuits or circuits containing some active elements with complicated models. An analysis of circuits with real parameters being modeled is a typical task for simulation programs. Programs for symbolic analysis with subsequent numerical analysis, for example SNAP, seem to be excellent tools for the analysis of s -domain circuit functions.

The choice of the type of MNA for hand-and-paper analysis is a trade-off between the comfort of constructing the set of circuit equations and its size, which determines the computational effort. The advantage of the method of stamps, widely used in most simulation programs, consists in the simple algorithm of writing the equations in the computer memory, but at the cost of a huge number of these equations. On the other hand, the V/I method provides a minimum number of equations but with a more complicated way of their composition. In addition, the efficiency of this method is high only for limited types of circuits. The dead row method can be a good compromise, probably the best of all, for the analysis of circuits employing the finite-gain voltage amplifiers. It has been shown that it can be also useful for circuits utilizing OpAmps and current conveyors.

The circuits with magnetic couplings form a special group. The method of mesh currents can be useful here. However, it is not described in this Chapter because its algorithmization is not easy. Simple circuits containing ideal transformers can be sometimes solved intuitively only on the basis of transforming the relations between voltages and currents. Nevertheless, one possibility of analyzing such circuits *via* MNA is described in section 3.5. The number of equations is relatively high, which makes this method preferable in computer-aided analysis.

Acknowledgements

This work has been supported in part by the Czech Grant Agency under grants Nos. 102/08/0784, 102/09/1628, and by the research programmes of BUT No. MSM0021630503/513 and UD Brno No. MO FVT0000403, Czech Republic.

The research leading to these results has also received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 230126.

References

- [1] C-W. Ho, A.E. Ruehli, and P.A. Brennan, "The modified nodal approach to network analysis", *IEEE Transactions on Circuits and Systems*, vol. CAS-22, no. 6, pp. 504-509, June 1975.
- [2] A. Vladimirescu, *The SPICE Book*. John Wiley & Sons, Inc., 1994.
- [3] R. Kielkowski, *Inside SPICE*. McGraw-Hill, 1998.
- [4] J. Vlach, *Basic network theory with computer applications*. Van Nostrand Reinhold, New York, 1992.
- [5] J. Vlach, and K. Singhal, *Computer methods for circuit analysis and design*. Van Nostrand Reinhold Company, New York, 2nd edition, 1994.
- [6] A. Luchetta, and S. Manetti, "SAPWIN – A Symbolic simulator as a support in electrical engineering education", *IEEE Transactions on Education*, vol. 44, no. 2, p. 213, May 2001.
- [7] Z. Kolka, D. Biolek, and V. Biolkova, "Symbolic analysis of linear circuits with modern active Elements", *WSEAS Transactions on Electronics*, vol. 5, no 6, pp. 88-96, 2008.
- [8] H. Walscharts, G. Gielen and W. Sansen, "Symbolic simulation of analog circuits in s- and z-domain", in *IEEE International Symposium on Circuits and Systems*, 1989, pp. 814-817.
- [9] G. Gielen, H. Walscharts and W. Sansen, "ISAAC: A Symbolic simulator for analog integrated circuits", *IEEE Journal of Solid-State Circuits*, vol. SC-24, no.6, pp. 1587-1597, December 1989.
- [10] D. Biolek, V. Biolková and K. Vrba, "Teaching linear circuits analysis effectively", in *4th UICEE Annual Conference on Engineering Education*, Bangkok, Thailand, 7-10 February 2001, pp. 277-280.
- [11] D. Biolek, *Solving electronic circuits. The book about their analysis*. BEN, 2003.
- [12] D. Biolek, R. Senani, V. Biolkova and Z. Kolka, "Active elements for analog signal processing: classification, review, and new proposals", *Radioengineering*, vol. 17, no. 4, pp. 15-32, December 2008.

- [13] G. Massobrio and P. Antognetti, *Semiconductor device modeling with SPICE*. McGraw-Hill, 1993.
- [14] E. Tlelo-Cuautle and M.A. Duarte-Villasenor, *Evolutionary electronics: automatic synthesis of analog circuits by GAs*. Success in Evolutionary Computation, Studies in Computational Intelligence (SCI), vol. 92, pp. 165-187, Springer Berlin/Heidelberg, 2008.
- [15] The website of SNAP program. Available: <http://snap.webpark.cz/indexa.html> [Accessed May 10, 2010].

CHAPTER 3**Modeling Active Devices with Nullors for Analog Signal Processing****Carlos Sánchez-López****Autonomous University of Tlaxcala, México and IMSE-CNM, CSIC and University of Sevilla, Spain*

Abstract: This chapter describes the modeling of nullor-based active devices from the circuit level of abstraction. After a brief overview on the nullor concept and its properties, the modeling of active devices not only at the voltage-mode but also at the current-mode and the mixed-mode of operation from two-port and four-terminal network point of view is described in some detail. An important view that permeates the chapter is that the nullor-based models are not too complex and they can be introduced in CAD tools. Furthermore, parasitic elements can easily be added in order to predict their impact on the final response of the circuit. Several examples using nullor-based models illustrate its use to calculate fully-symbolic small-signal characteristics of linear or linearized analog circuits.

Keywords: Symbolic analysis, nodal analysis, modified nodal analysis, nullor, pathological elements, operational amplifiers, current mirrors, analog signal processing, controlled sources, stamps.

1. Introduction

It is well known that the behavior of all the active devices has mainly been modeled with Voltage-Controlled Voltage Sources (VCVS), Voltage-Controlled Current Sources (VCCS), Current-Controlled Voltage Sources (CCVS) or Current-Controlled Current Source (CCCS), and they are often used in CAD tools, such as Hspice, where very accurate models for a large variety of active devices are provided [1-3]. These models are widely used to formulate the system of equations of analog networks by using several methods, such as: the Nodal Analysis (NA) method, the Modified Nodal Analysis (MNA) method¹ or Tableau method.

From the symbolic analysis techniques point of view, the NA has widely been used to describe networks and it is well known that only NA-compatible elements can directly be introduced into the admittance matrix by using the element stamp method. This disadvantage has been overcome by using the MNA method, in which additional columns and rows are incorporated into the admittance matrix and the non-NA compatible elements are readily included by using stamps [1-3]. However, a disadvantage on the use of controlled sources to model active devices is that the admittance matrix becomes large, since it depends on the number of node voltages and of the branch currents associated to the type of element contained in the circuit.

On the one hand, the use of very accurate models represents a computational effort higher than by using compact models during circuit analysis. Also, as an analog designer deals with simplified analytical equations, then compact models are quite useful to enhance the modeling and simulation time during the design process. In this sense, the nullor element has demonstrated to be efficient to model the behavior of active devices [4-9]. Furthermore, in a nullor-based model, the parasitic elements

*Address correspondence to Carlos Sánchez-López: Autonomous University of Tlaxcala, México and IMSE-CNM, CSIC and University of Sevilla, Spain; E-mail: carlsanmx@yahoo.com.mx

¹See Chapter 2 for a detailed description on the NA method, the MNA method and its modifications.

can easily be added in order to predict their impact on the final response of the circuit. As a consequence, the main advantage of using nullors, is that one can describe a circuit with only two-terminal elements such as: resistors, capacitors, nullators, norators and independent current sources. Most important, is that all the controlled sources can be modeled by using nullors. Hence, the standard NA method can be used to formulate the system of equations, which models the behavior of a nullor-based equivalent circuit. Henceforth, the nullor is used herein to model the behavior of active devices, but taking into account that a circuit should be modeled in the simplest possible way, so that its impact on overall simulation accuracy must be tolerable. In this manner, this chapter describes the issues concerning the small-signal modeling of active devices by using nullors.

2. Nullor Concept

The nullator and norator are theoretical active devices that have been used in the analysis, design and synthesis of linear or linearized circuits [6-13]. Since Tellegen exposed the concept of ideal operational amplifier, the nullor concept was also introduced indirectly [5]. Later, Carlin, in 1964, made an attempt to use both nullators and norators as single active devices, but taking into account that they cannot be physically realizable [4]. Tellegen also explained that these singular elements can be considered only as mathematical concepts without a physical content. Again, Carlin proposed the combination of the nullator and the norator, which resulted in a useful physical device, the *nullor* [4, 5]. The nullor consists of a nullator and a norator, as shown in **Fig. 1**, and its behavior is distinguished by very special voltage-current relationships [10].

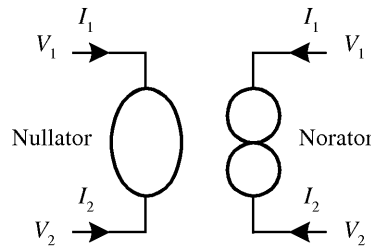


Figure 1: Nullor symbol.

The input port of the nullor is modeled by the nullator which is characterized by two equations:

$$V_2 = V_1 = \text{arbitrary}, \quad I_2 = I_1 = 0 \quad (1)$$

So, the nullator is simultaneously one open- and short-circuit. The output port of the nullor is modeled by the norator where both, the voltage and current can be assumed to have arbitrary values:

$$V_2 \neq V_1 = \text{arbitrary}, \quad I_2 = -I_1 = \text{arbitrary} \quad (2)$$

The nullor is a two-port element and it is known as *universal active element* [10]. This concept means that the nullor along with capacitors and resistors can be used to design a maximum number of functions with the minimum number of active devices. That is, if a suitable set of linear and nonlinear passive elements is available, then no

active element other than nullors are needed to implement any linear or nonlinear circuit function. So, nullators, norators, resistances, along with capacitances can synthesize a complete set of linear or linearized equations.

3. Nullor Equivalences

Since its introduction, the nullor has been found useful in the analysis, modeling and synthesis of analog circuits. As a consequence, several papers have been reported in the literature on the use of methods or computational algorithms based on nullor-elements for the synthesis and modeling of active devices [6-13]. Because any analog network can be modeled with nullators, norators and impedances, it is worth mentioning some equivalence between them, since they can be very useful in the synthesis and modeling process. Thus, the main equivalences of the combinations between nullators and norators for networks that contain nullors and impedances are shown in **Fig. 2**. For instance, in **Fig. 2(a)** a current flowing from node a , cannot flow from a to b , since the current through the nullator is zero, so that a series connection of the nullator and norator is equivalent to one open-circuit. In **Fig. 2(b)**, the current can flow from a to b through the norator, also the voltage across a and b becomes zero from the property of the nullator, so that a parallel connection of the nullator and norator is equivalent to one short-circuit. The remaining connections have equivalents according to the nullator and norator $i-v$ characteristics.

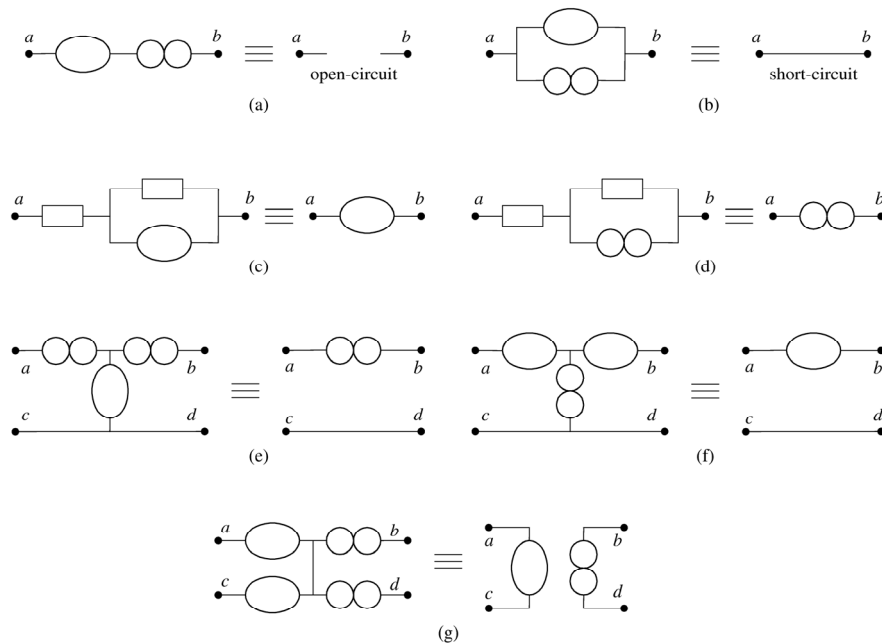


Figure 2: Nullator and norator equivalences.

On the other hand, the nullor along with grounded resistors can be manipulated in order to obtain inverting properties, features that the nullator and the norator cannot model by themselves [14]. The main purpose of the introduction of inverting properties is that the behavior of some active devices involves inverting the voltage and current input-signals. In this sense, the Current-Mirror (CM) and the Voltage-Mirror (VM), both as active devices, can perform this task and their behavior also should be modeled with the nullor. Thus, by manipulating the nullor along with grounded resistors, the behavior of a CM and a VM, both with ideal unity-gain can easily be obtained, as shown in **Fig. 3** [14].

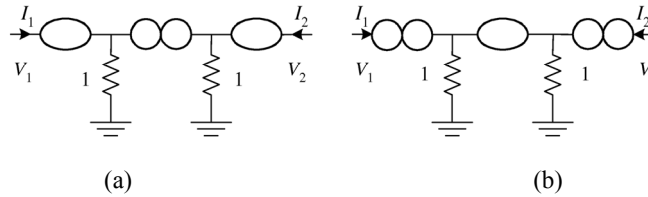


Figure 3: Nullor and grounded resistor-based VM (a) and CM (b).

Therefore, by analyzing the equivalent circuits, one can see that the VM, shown in **Fig. 3(a)**, is characterized by:

$$V_2 = -V_1 = \text{arbitrary}, \quad I_2 = I_1 = 0 \quad (3)$$

and the CM, shown in **Fig. 3(b)**, is characterized by:

$$V_2 \neq V_1 = \text{arbitrary}, \quad I_2 = I_1 = \text{arbitrary} \quad (4)$$

So, the inverting behavior of the nullator and norator is achieved. In [15], the nullor-based models of the VM and CM include parasitic elements. In the same manner as for the nullor, equivalences between the combinations of nullators, norators, CMs, VMs and impedances within an analog network can be obtained. Note, however, if the V_1 - or V_2 -terminal from **Fig. 3(a)** is grounded and by applying the equivalences shown in **Fig. 2**, the VM is reduced to a nullator. In the same manner, if any terminal in **Fig. 3(b)** is grounded and after by applying the equivalences shown in **Fig. 2**, then a norator is obtained.

4. Nullor-Based Active Device Models

In this section, the derivation of nullor-based active device models will be introduced. First, ideal models of active devices are derived and later on, parasitic elements are included in order to obtain a more complete model. Thus, these nullor-based models are the basis to achieve a fully-symbolic analysis of analog networks by applying the standard NA method.

4.1. Independent voltage source and controlled sources

An analog network is often composed of several two-terminal elements, namely: resistors, capacitors, inductors, independent voltage sources, independent current sources and the four basic controlled sources. By using the nullor element, the behavior of certain elements can be approximated by combining them in appropriate configurations [6, 7]. Thus, an independent voltage source can be modeled by using the nullor element as shown in **Fig. 4**.

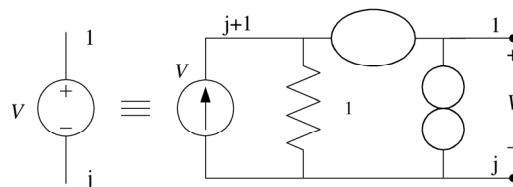


Figure 4: Nullor-based independent voltage source.

The analysis of the independent voltage source is as follows: the current can only flow through the resistor, thus, the current is transformed to voltage and by

considering the two constraints of the nullator, the voltage is obtained in the output node. In the same way, the four controlled sources can ideally be modeled as shown in Fig. 5, and these models result by applying the properties and equivalences of the nullor shown in Fig. 2, in order to model the behavior of the initial circuit.

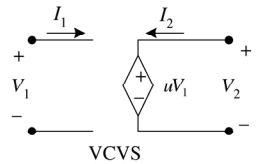
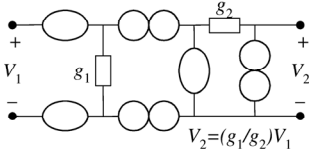
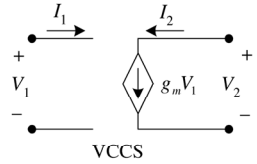
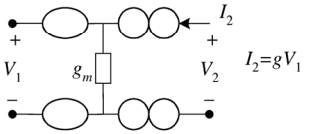
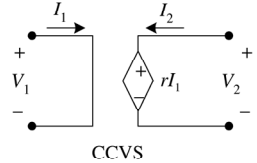
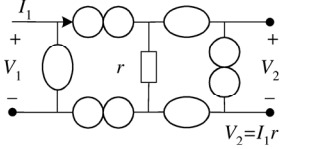
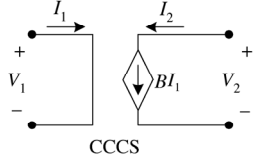
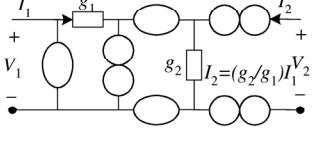
Symbol	Definitions	Equivalent Nullor
 <p>VCVS</p>	$\begin{bmatrix} V_1 \\ I_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ u & 0 \end{bmatrix} \begin{bmatrix} V_2 \\ -I_2 \end{bmatrix}$	 <p>$V_2 = (g_1/g_2)V_1$</p>
 <p>VCCS</p>	$\begin{bmatrix} V_1 \\ I_1 \end{bmatrix} = \begin{bmatrix} 0 & -1 \\ 0 & g_m \end{bmatrix} \begin{bmatrix} V_2 \\ -I_2 \end{bmatrix}$	 <p>$I_2 = gV_1$</p>
 <p>CCVS</p>	$\begin{bmatrix} V_1 \\ I_1 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ r & 0 \end{bmatrix} \begin{bmatrix} V_2 \\ -I_2 \end{bmatrix}$	 <p>$V_2 = I_1 r$</p>
 <p>CCCS</p>	$\begin{bmatrix} V_1 \\ I_1 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & -1/B \end{bmatrix} \begin{bmatrix} V_2 \\ -I_2 \end{bmatrix}$	 <p>$I_2 = (g_2/g_1)I_1V_2$</p>

Figure 5: Controlled sources, definitions and nullor-based models.

4.2. Operational amplifiers

Nowadays, there are a large variety of active devices that have been proposed in the literature for analog signal processing [16, 17]. All the active devices can be grouped in voltage-, current- and mixed-mode or according to the type of signal that can be processed in their input-output terminals. Without losing generality, a four-terminal analog circuit can be described as a two-port network under the condition $i_a = -i_b$ and $i_b = -i_a$ as shown in Fig. 6(a). Furthermore, if any circuit can be described as a two-port network, then the impedance levels at each port can ideally be considered as very high or very low. From the voltage-current relationships of the nullor, a floating nullator can model a node with high impedance, or a node with low-impedance if any terminal of the nullator is grounded [11, 12]. Contrarily, the norator can model both impedance levels: high or low depending of the signal to be measured.

As a result, not only the impedance levels but also the gain-equations of the operational amplifiers should be considered in order to generate their nullor-based models. Thus, by applying the nullor properties, the behavior of some active devices that can process only a type of signal in their input-output terminals, such as: the Operational Amplifier (OpAmp), the Operational Transconductance Amplifier (OTA), the Operational Transresistance Amplifier (OTRA) and the Current Operational Amplifier (COA) can adequately be modeled as a two-port network.

On the other hand, when the behavior of an analog circuit is described by a four-terminal network, as shown in **Fig. 6(b)**, the impedance levels in the input-terminals, Z_a and Z_b , and in the output-terminals, Z_c and Z_d can be very high or very low. Hence, the behavior of some operational amplifiers cannot be modeled directly by using the two-port concept and they are better modeled as a four-terminal network [10]. In this case, the nullor can also be used to model the behavior of operational amplifiers with features of a four-terminal network, but by considering the impedance level and the input-output signal associated to each terminal. Instead, if the impedance level in the two input-terminals is high and further, the impedance level for each output-terminal is high and low, then the two input-terminals can be modeled as one-port and the output port as two-terminals. For instance, the Operational Floating Amplifier (OFA) has two input-terminals with an impedance level high, which can be modeled as one-port. For the output terminals, the impedance levels are high and low in agreement to the kind of output signal, and are better modeled as two-terminals [18].

Likewise, for those operational amplifiers that only allow current-mode signals at their input-terminals, such as: the Floating Operational Transresistance Amplifier (FOTRA), the Current Feedback Operational Amplifier (CFOA), the Current Feedback Operational Transconductance Amplifier (CFB-OTA), COA and OTRA, it is necessary to invert the current signal at their negative terminals. Therefore, the nullor-based model of the CM, shown in **Fig. 3(b)**, can be used. Maybe, one could contemplate the use of the CCCS or CCVS to model those operational amplifiers mentioned above. However, this is not feasible, since not only complex nullor-based models are obtained but also, the models shown in **Fig. 5** have a serious disadvantage, since they are only considering one input-current, limiting their application to specific circuits, where the input-current is flowing in one direction. In this manner, the nine operational amplifiers along with their nullor-based models are shown in **Fig. 7**. One important observation is that into the nullor-based models, only grounded resistors are used, hence, they have only one entry in the NA matrix.

4.3. Current conveyors

The current conveyor is considered as a universal analog building block designed for voltage-, current- or mixed-mode signal processing [19-22]. A review of the state-of-the-art of current-mode circuits has been introduced in [23]. There are several families of current conveyors such as: inverting or non-inverting positive/negative-type first generation Current Conveyors (CCI+, CCI-, ICC+, ICCI-), inverting or non-inverting positive/negative-type second generation Current Conveyors (CCII+, CCII-, ICCII+, ICCII-), inverting or non-inverting positive/negative-type third generation Current Conveyors (CCIII+, CCIII-, ICCIII+, ICCIII-), all with a single or Multiple Outputs (MO) [13]. Among all the current conveyors, the inverting and non-inverting CCII±s are the most versatile and they have widely been used to design linear and nonlinear circuits [24].

In the same manner that operational amplifiers, nullor-based models of current conveyors can also be generated by considering the impedance levels associated to their input-output terminals and constitutive equations. For instance, let us consider the ICCII+ shown in **Fig. 8(a)** as a three-terminal active device, since the impedance levels at the input-output terminals are low and high, respectively. The operation of the ICCII+ is characterized by the following matrix form [24]:

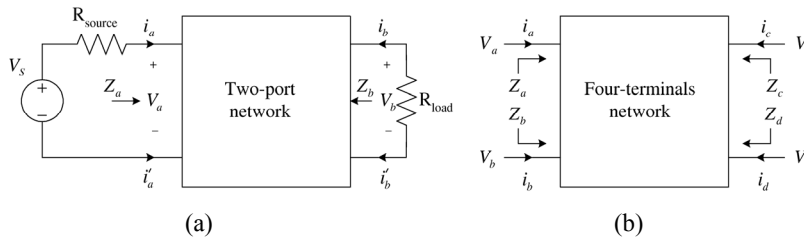


Figure 6: (a) Two-port network and (b) four-terminal network.

Active Device	Gain Equation	Impedance Level	Nullor-based Model	Name
	$V_{out} = A_v(V^+ - V^-)$	$Z_{in} = High$ $Z_{out} = Low$		Operational Amplifier (Opamp)
	$I_{out} = g_m(V^+ - V^-)$	$Z_{in} = High$ $Z_{out} = High$		Operational Transconductance Amplifier (OTA)
	$V_w = A_v(V^+ - V^-)$ $I_z = A_f I_w$	$Z_{in} = High$ $Z_w = Low$ $Z_z = High$		Operational Floating Amplifier (OFA)
	$V_{out} = R_m(I_p - I_n)$	$Z_{in} = Low$ $Z_{out} = Low$		Operational Transresistance Amplifier (OTRA)
	$I_{out} = A_f(I_p - I_n)$	$Z_{in} = Low$ $Z_{out} = High$		Current Operational Amplifier (COA)
	$V_w = R_m(I_p - I_n)$ $I_z = A_f I_w$	$Z_{in} = Low$ $Z_w = Low$ $Z_z = High$		Floating OTRA (FOTRA)
	$V_{out} = R_m I_a$ $V^+ = V^-$	$Z^+ = High$ $Z = Low$ $Z_{out} = Low$		Current Feedback Operational Amplifier (CFOA)
	$I_{out} = A_f I_a$ $V^+ = V^-$	$Z^+ = High$ $Z = Low$ $Z_{out} = High$		Current Feedback OTA (CFB OTA)
	$V_w = R_m I_a$ $V^+ = V^-$ $I_z = A_f I_w$	$Z^+ = High$ $Z = Low$ $Z_z = High$ $Z_w = Low$		Operational Floating Conveyor (OFC)

Figure 7: Nullor-based operational amplifiers.

$$\begin{bmatrix} I_y \\ V_x \\ I_z \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} V_y \\ I_x \\ V_z \end{bmatrix} \quad (5)$$

From (5), one can see that the equality $V_x = -V_y$ can be modeled with a VM, as shown in **Fig. 3(a)**, and $I_z = I_x$ can be modeled with a CM, as shown in **Fig. 3(b)**. Besides, the impedance levels from the original circuit still hold. Thus, the nullor-based model of the ICCII+ is generated and shown in **Fig. 8(b)**. The same methodology is used in order to generate the nullor-based models of the other current conveyors, as depicted in **Figs. 9, 10** and **11**. Here, it is worth mentioning that for a physical network, which is modeled with nullors, it should have the same number of nullators and norators, in order to satisfy the branch voltage-current relationships.

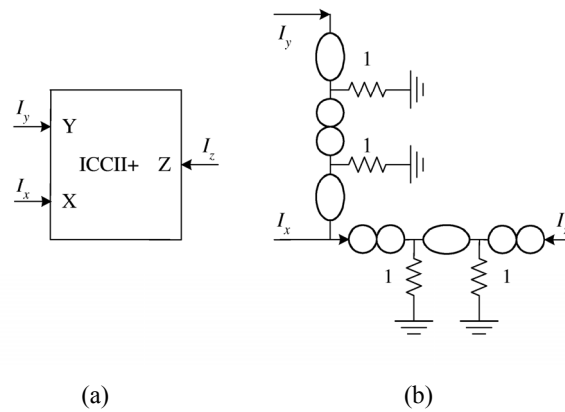


Figure 8: (a) ICCII+ symbol (b) nullor-based model.

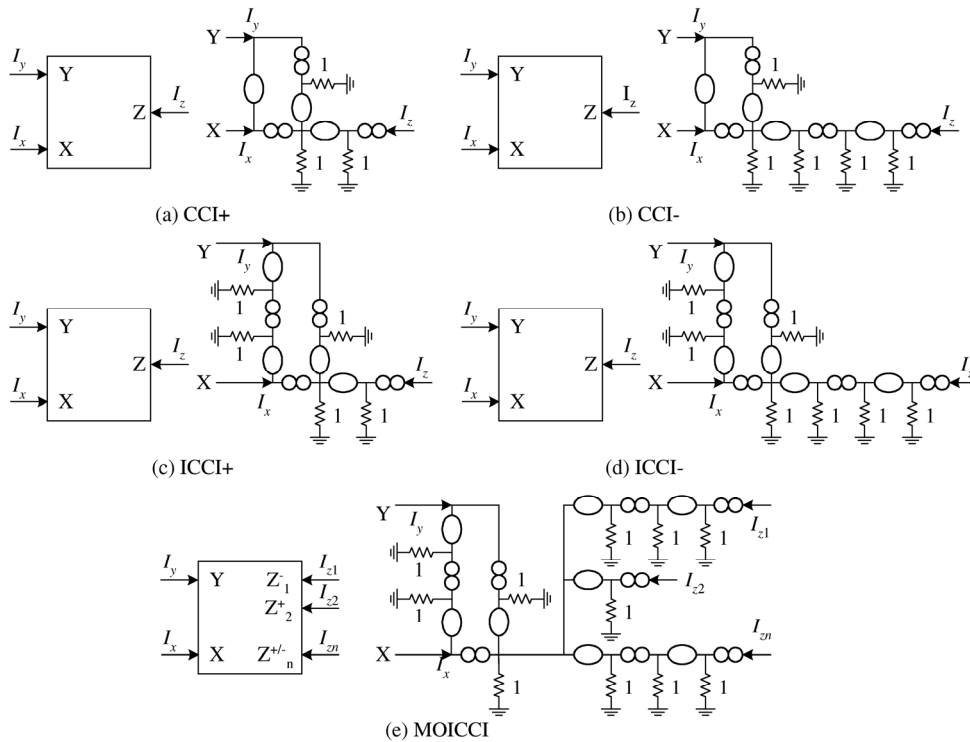


Figure 9: Nullor-based (MO) (I) CCI± models.

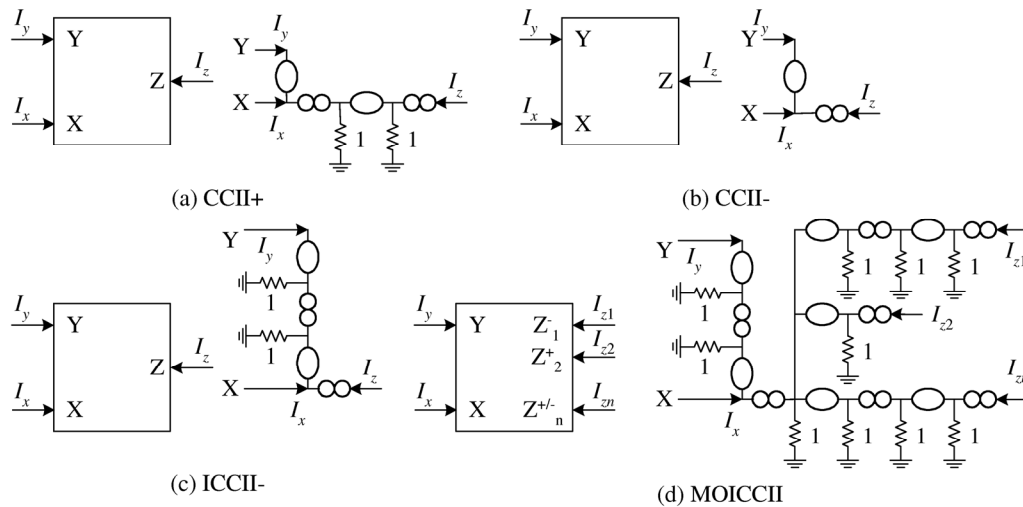


Figure 10: Nullor-based (MO) (I) CCII± models.

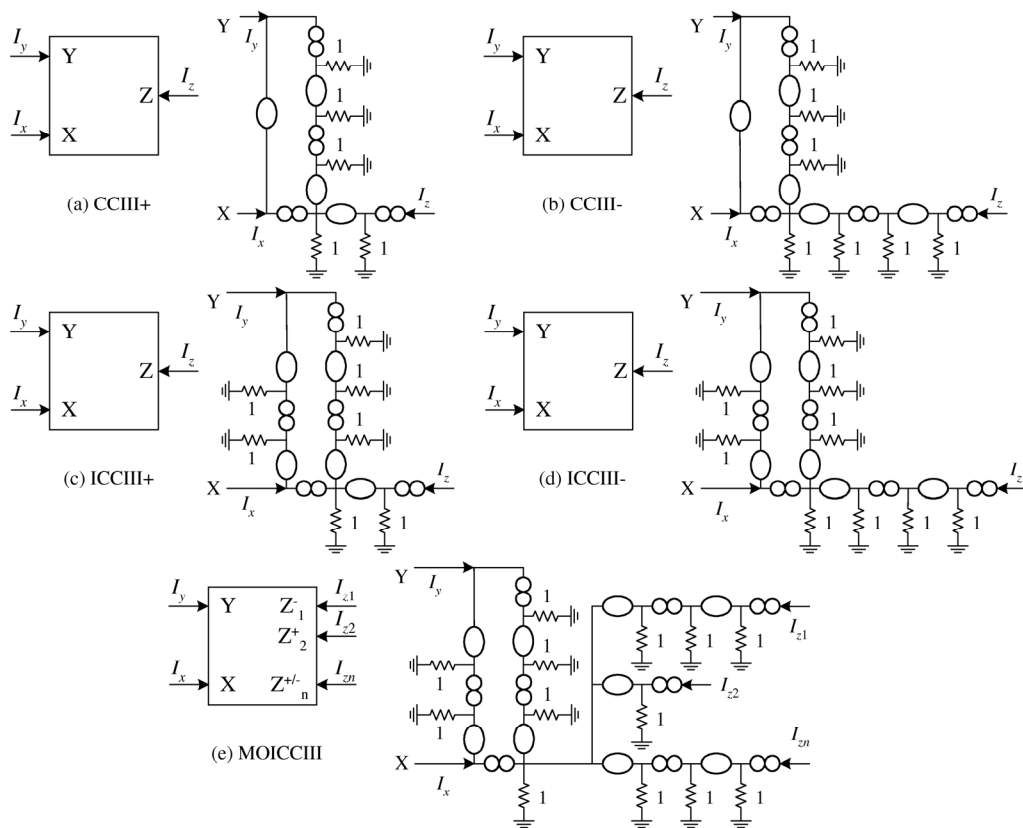


Figure 11: Nullor-based (MO) (I) CCIII±.

4.4. Other active devices

In the literature other versatile active building blocks with features that involve differential signals have been reported, such as: the Current Differencing Unit (CDU), the Current Differencing Buffered Amplifier (CDBA), the Differential Voltage Current Conveyor (DVCCII), the Differential Current Voltage Conveyor (DCVC), the Fully-Differential Current Conveyor (FDCCII) and the Differential Difference Amplifier (DDA), among others. An excellent survey of well known and new active

devices is given in [16]. Several applications based on using these active devices have been introduced in the literature, where active devices are modeled with controlled sources [16]. On the other hand, to accomplish efficiently a fully-symbolic analysis of analog networks by applying only the standard NA method, it is necessary to be able to obtain suitable representations for such active devices by using nullor elements. Here we will show the generation of the nullor-based model of the DVCCII with double output, as shown in **Fig. 12(a)** [25]. The DVCCII is a five-terminal active device that is defined by the following matrix equation:

$$\begin{bmatrix} V_x \\ I_{y1} \\ I_{y2} \\ I_{z1} \\ I_{z2} \end{bmatrix} = \begin{bmatrix} 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} I_x \\ V_{y1} \\ V_{y2} \\ V_{z1} \\ V_{z2} \end{bmatrix} \quad (6)$$

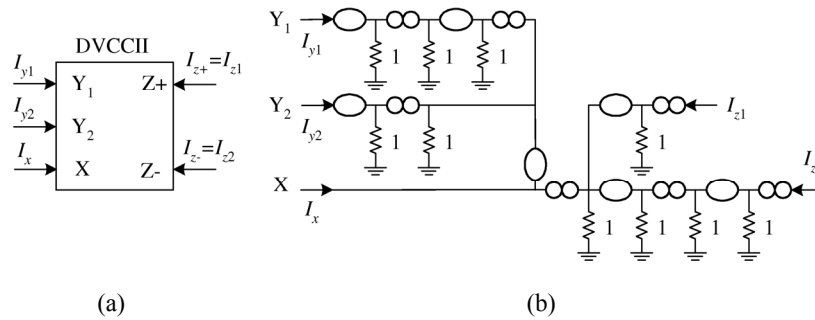


Figure 12: DVCCII with double output (a) symbol (b) nullor-based model equivalent.

According to (6), the differential voltage given as: $V_x=V_{y1}-V_{y2}$, can be modeled by using nullor and grounded resistors. The reason on the use of grounded resistors is that along with nullators and norators are used to transform a current to voltage or vice-versa. Therefore, to achieve the behavior of a differential voltage, VMs or CMs can be employed and the high impedance levels at the input-terminal still hold. On the other hand, the current behavior from I_x to $I_{z\pm}$, is similar to a CCII with double output, as shown in **Fig. 10(d)**. In this manner, the nullor-based model of the DVCCII with double output is shown in **Fig. 12(b)**.

4.5. Including parasitic elements in the nullor-based models

All the nullor-based models introduced in the previous subsections have been generated taking into account the ideal behavior of the active devices. However, when active devices are designed physically, the input-output resistance and capacitance levels, gain, input offset voltage or current and the frequency response are all finite. Therefore, it is widely necessary to include these effects in the nullor-based models; as well one can know the real limitations of an analog network, when the active devices are used in applications of analog signal processing. Here, we will only show the inclusion of the finite parasitic impedances associated to the input-output terminals of an active device. With this choice, parasitic resistors and capacitors can easily be included in the input-output terminals of the nullor-based models shown in **Figs. 7 to 12**, but by considering the type of signal that can be processed in their terminals. For instance, a CFOA is characterized by R_x in the x -terminal, R_y and C_y in the y -terminal, R_z and C_z in the z -terminal (here, R_m is the parallel of R_z and C_z and it

also represents the dominant pole, *i.e.* the bandwidth). Therefore, one more realistic model can be achieved, as shown in **Fig. 13**. Note that a CM is used to invert the direction of the current-output instead the CFOA model shown in **Fig. 7**. The same methodology can be used to obtain more realistic models for the other active devices.

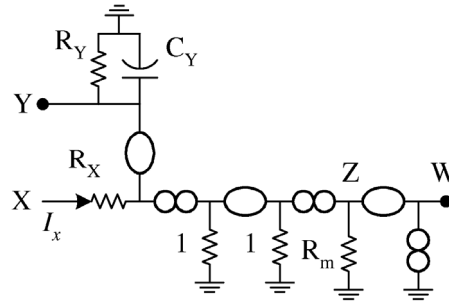


Figure 13: Nullor-based model CFOA including parasitic elements.

5. Formulation of the System of Equations

In this section, we present a novel method to formulate the system of equations from analog circuits, where the nullor-based models of the active devices generated in the previous section are used [26, 27]. As mentioned already, any analog network is composed by resistors, capacitors, independent voltage or current sources, controlled sources and active devices. Later on, if each independent voltage source, controlled source and active device is modeled with nullors, the original analog network is transformed to a nullor equivalent circuit, which is composed by nullators, norators, admittances and independent current sources. As a consequence, the standard NA method can be used to formulate the system of equations, which is expressed as:

$$J = Y_n V \tag{7}$$

where Y_n represents the admittance matrix, J is the vector of current sources and V is the vector of voltage-variables. It is worth mentioning that the nullator and norator properties should be applied in order to reduce the size of the admittance matrix [26, 27]. Indeed, the nodes of a nullator are related with the columns of the admittance matrix, thus, two columns in Y_n must be added according to the nodes of a nullator connected into the circuit and a column should be deleted if any terminal of the nullator is grounded. In the same manner for the norators; a row of the admittance matrix is deleted if any terminal of a norator is grounded and two rows in Y_n must be added according to the nodes of a norator connected in the circuit. This concept is illustrated in **Fig. 14**.

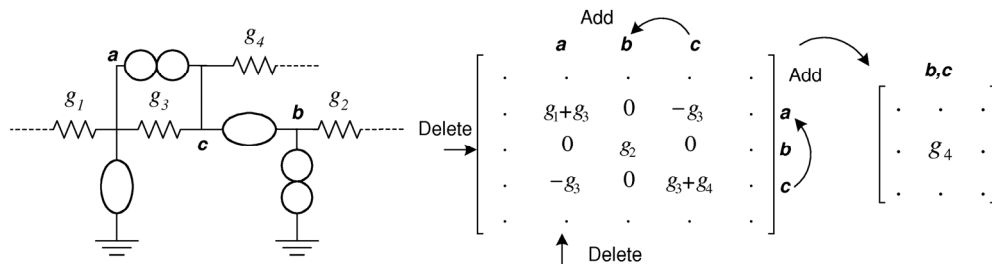


Figure 14: Reduction of the admittance matrix by applying the nullator and norator properties.

On the one hand, the formulation method described above is very useful when the symbolic analysis of analog circuits is done by paper and pencil. On the other hand, the formulation method of the system of equations has been improved in order to be quickly included in symbolic analyzers, which is based on the manipulation of the relationships of the admittances, nullators and norators [26, 27]. The formulation method is described below:

5.1. Data-structures of two-terminal elements

All the elements of an analog network are stored into an Hspice-like netlist. Thus, each independent voltage source and active device should be modeled by their nullor-based models, as shown in Fig. 4, 7 and 13. According to Fig. 15, the generation of data-structures for each two-terminal element is summarized in the following points:

1. Obtain the nodes from the netlist and store into a set of nodes called *Set-node*. The nodes are ordered in ascending form. The node labeled by 0 is the reference node and it is not included into *Set-node*.
2. Each active device is modeled with its nullor-based model and stored into a list of data-structures.

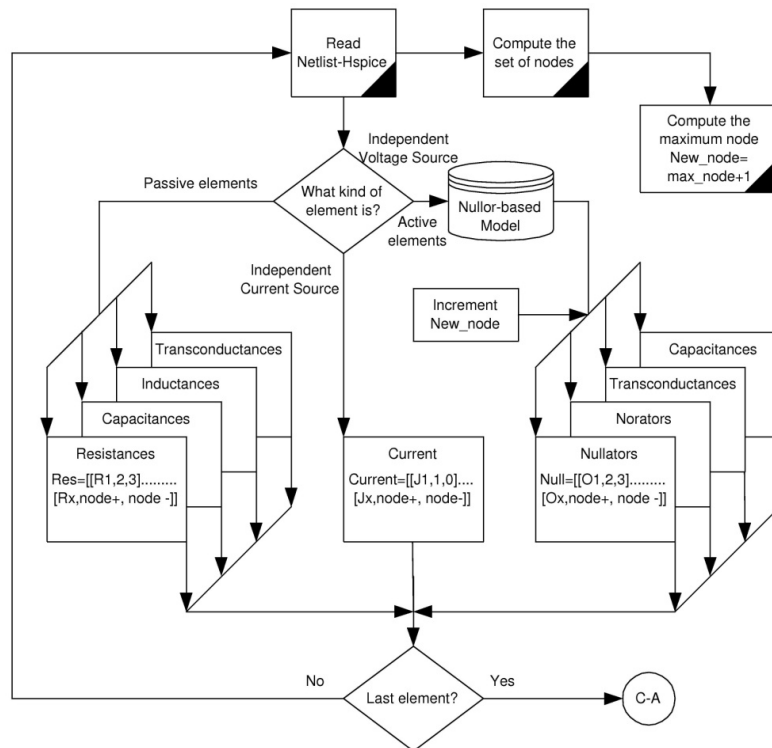


Figure 15: Data-structures generation.

5.2. Computing norator-nullator indexes

The nullators and norators should be manipulated to generate two vectors, namely: P (norator vector) and O (nullator vector), as shown in Fig. 16. These vectors have the indexes associated to the columns and rows variables of the admittance matrix and they are also used to fill the J and V vectors of the system of equations. The procedure to compute the indexes is as follows:

1. Read the pair of nodes of each norator and nullator.
 - If any node of a nullator or norator is equal to the reference node, skip to the next element.
2. Compare the pair of nodes of a norator or nullator with each subset of nodes stored previously.
 - If the nodes match, include the pair of nodes into the corresponding subset, which is ordered in ascending form.
 - If the nodes are not match, a new subset of nodes is formed and ordered in ascending form, and must be included into the P or O vector, respectively.
3. Compare each node of $Set-node$ with the first node of each subset of nodes of the P and O vector.
 - If the nodes match, the subset of nodes of the P or O vector should be reordered according to the position of the node in $Set-node$.
 - If the nodes are not matching, include the node of $Set-node$ into the P or O vector and placed in the same position as in $Set-node$. As a consequence, the final vectors are given as: $O = [\{O_1\}, \{O_2\}, \dots, \{O_x\}]$ and $P = [\{P_1\}, \{P_2\}, \dots, \{P_x\}]$, where $O_x = \{a_1, a_2, \dots, a_n\}$ is a subset of nodes of the O vector and $P_x = \{b_1, b_2, \dots, b_n\}$ is a subset of nodes of the P vector and a_n along with b_n are the nodes of the subsets.

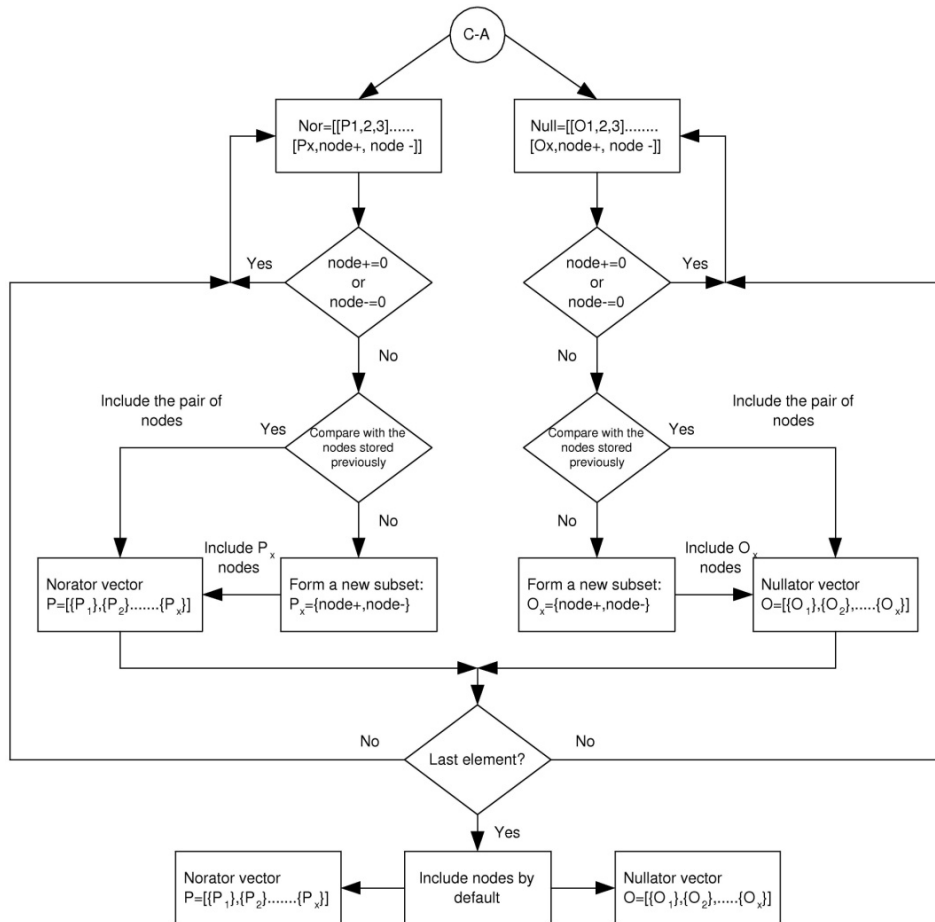


Figure 16: Nullator and norator indexes generation.

5.3. Building the nodal admittance matrix

According to **Fig. 17**, the nodal admittance matrix is obtained by manipulating the admittances, nullators and norators data-structures as follows:

1. Compare each node of O_x with the pair of nodes associated to admittances.
 - If the node from O_x match with any node of some floating admittances, include the pair of nodes along with the symbol of each admittance into a list called $Col_{\{O_x\}} = [(a_n, [node_{\pm}, elem_n])]$, where O_x represent the nodes of the j -th column, a_n represent the nodes of O_x , $node_{\pm}$ is the node not-matching of the admittance and $elem_n$ is the name of the n -th admittance.
 - If the node from O_x match with the node of some grounded admittances, include the node along with the name of the admittance as: $Col_{\{O_x\}} = [(a_n, elem_n)]$.

These lists are very important, because each list $Col_{\{O_x\}}$ has only the elements of each column of the admittance matrix. In order to formulate the nodal admittance matrix, the nodes of the norator vector should be used to obtain the coefficients of the Y_n matrix, where each P_x represent the nodes of the i -th row.

2. Compare each node of P_x with the nodes of each list $Col_{\{O_x\}}$.
 - If $b_n = a_n$, all the admittances of $Col_{\{O_x\}}$ are added in Y_{ij} with positive sign.
 - If $b_n = node_{\pm}$, only the admittance associated to $node_{\pm}$ is added in Y_{ij} with negative sign.

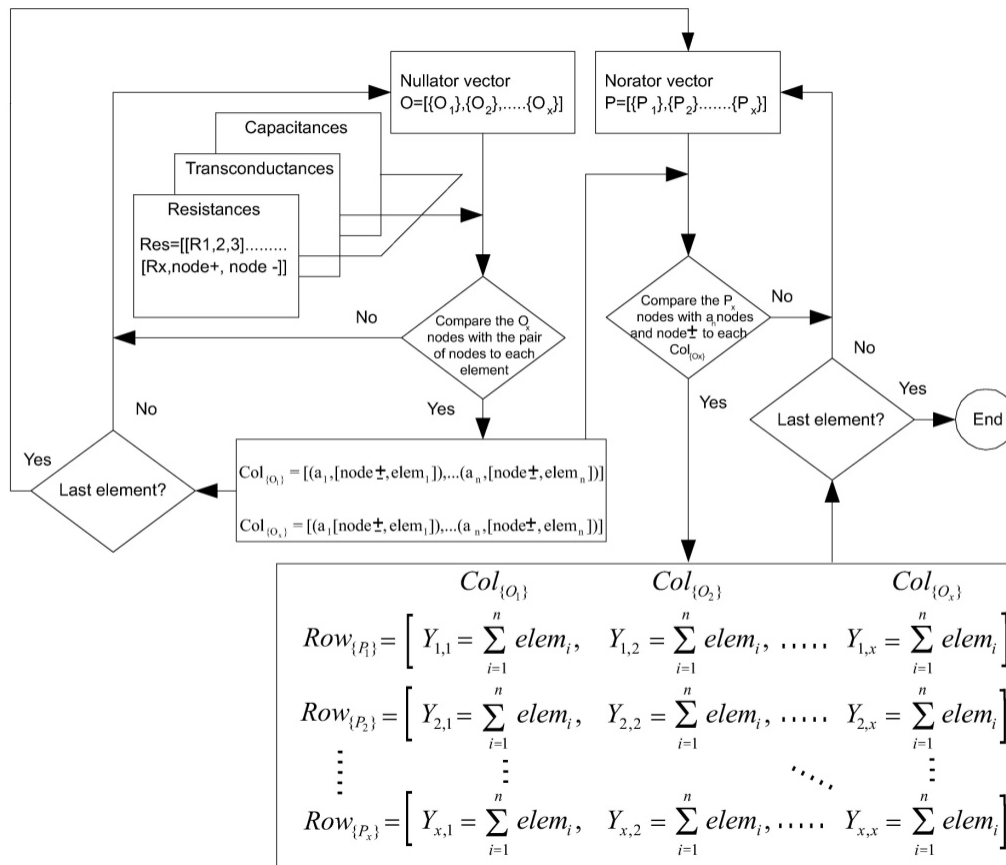


Figure 17: Diagram of flow to obtain the nodal admittance matrix.

5.4. Generating the V and J vectors

In the nullator vector, the nodes of O_x are the nodes of the nullor equivalent circuit with the same potential. Therefore, each subset of nodes O_x , represent a nodal voltage in the admittance matrix. Thus, the voltage vector is obtained from the nullator vector and given as:

$$V = [V_{\{o_1\}}, V_{\{o_2\}}, \dots, V_{\{o_x\}}]^T \quad (8)$$

In the norator vector, the nodes of P_x represent an entry of a current source given as:

$$J = [P_1, P_2, \dots, P_x]^T \quad (9)$$

To fill (9), the nodes of P_x must be compared with the nodes (k, l) of all the current sources:

- If $b_x=k$ the current source is added in (9) with negative sign, according to the position of P_x .
- If $b_x=l$ the current source is added in (9) with positive sign, according to the position of P_x .

Hence, for any analog circuit where the active device is modeled with nullor, the nullor equivalent circuit has n nodes and m nullors, thus, the size of the admittance matrix is equal to $[n-m] \times [n-m]$. In order to compute fully-symbolic small-signal characteristics of analog circuits, the solution of the nodal admittance matrix can be performed by applying Cramer's rule or Determinant Decision Diagrams¹ [28, 29], for instance.

6. Illustrative Examples

In this section, we present some examples on symbolic analysis of analog circuits, in order to show the usefulness of the proposed nullor-based models and the potentiality of the proposed symbolic formulation method.

6.1. OTRA-based universal filter

As a first example, let us consider the universal filter taken from [30], which is depicted in **Fig. 18(a)**. Substituting the behavior of the OTRA and of the independent voltage source with their nullor-based models, as depicted in **Figs. 4** and **7**, it results in a nullor equivalent circuit, as shown in **Fig. 18(b)**.

From **Fig. 18(b)** one can see that the circuit is only composed by nullators, norators, capacitors, resistors and one independent current source. Hence, the standard NA method can be executed to formulate the system of equations, which is given by:

¹ See Chapter 8 for a detailed description on Determinant Decision Diagrams.

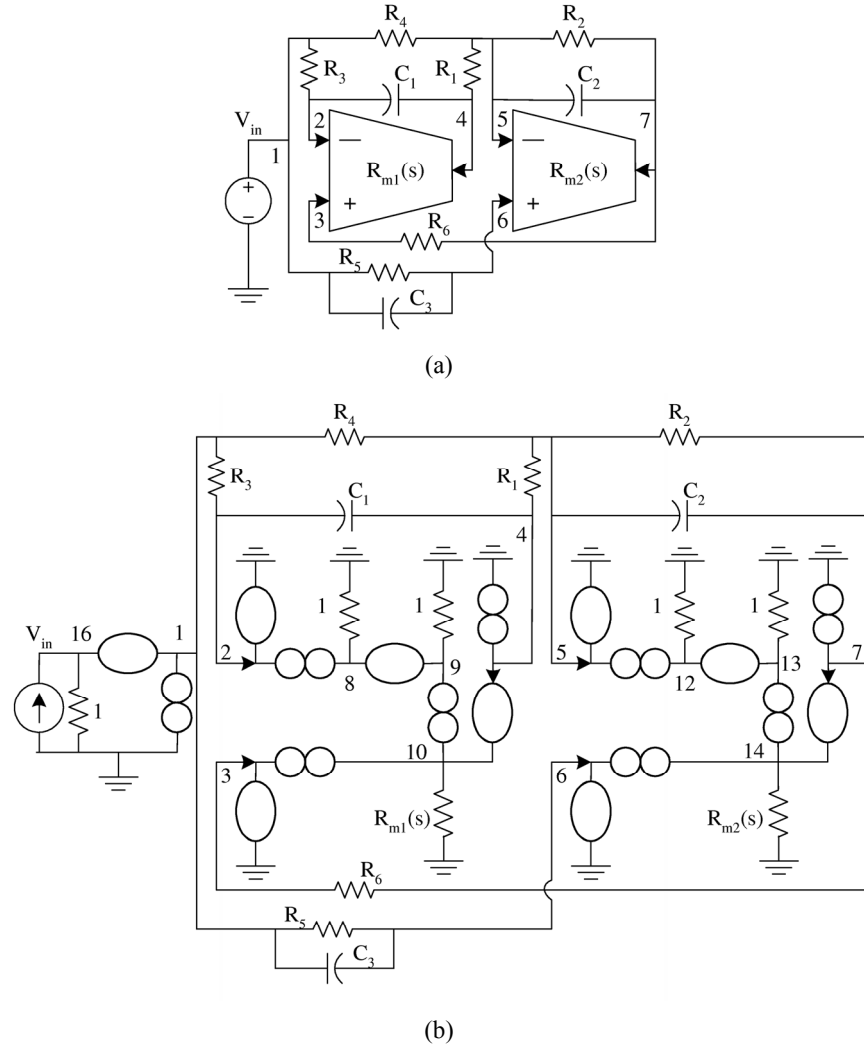


Figure 18: (a) OTRAs-based universal filter (b) nullor equivalent circuit.

To reduce the order of (10), the nullor properties are applied. Thus, from **Fig. 18(b)** and by using the nullator properties, the columns: 2, 3, 5 and 6 in (10) should be deleted. Also, the coefficients of some columns are added: 1 with 16, 4 with 10, 7 with 14, 8 with 9, and 12 with 13. By applying the norator properties, all the rows labeled as: 1, 4 and 7 should be deleted. In the same manner, some rows are added: 2 with 8, 3 with 9 and 10, 5 with 11, 6 with 12 and 13. Therefore, the admittance matrix is reduced as:

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ V_{in} \end{bmatrix} = \begin{bmatrix} -\frac{1}{R_3} & -sC_1 & 0 & 1 & 0 \\ 0 & -\frac{1}{R_{m1}(s)} & -\frac{1}{R_6} & 1 & 0 \\ -\frac{1}{R_4} & -\frac{1}{R_1} & -sC_2 - \frac{1}{R_2} & 0 & 1 \\ -sC_3 - \frac{1}{R_5} & 0 & \frac{1}{R_{m2}(s)} & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_{\{1,16\}} \\ V_{\{4,10\}} \\ V_{\{7,14\}} \\ V_{\{8,9\}} \\ V_{\{12,13\}} \end{bmatrix} \quad (11)$$

By resolving (11) to $V_{Out}=V_{\{7,14\}}$, the fully-symbolic transfer function is given by:

$$\frac{V_{Out}}{V_{in}} = \frac{\frac{C_3}{C_2} s^2 + \frac{1}{C_2} \left(\frac{C_3}{R_{m1}(s)C_1} + \frac{1}{R_5} - \frac{1}{R_4} \right) s + \frac{1}{R_{m1}(s)C_1C_2} \left(\frac{1}{R_5} - \frac{1}{R_4} \right) + \frac{1}{R_1R_3C_1C_2}}{s^2 + \left(\frac{1}{R_{m1}(s)C_1} + \frac{1}{R_{m2}(s)C_2} + \frac{1}{R_2C_2} \right) s + \frac{1}{R_{m1}(s)C_1C_2} \left(\frac{1}{R_2} + \frac{1}{R_{m2}(s)} \right) + \frac{1}{R_1R_6C_1C_2}} \quad (12)$$

Since the transresistance gain of the OTRA is finite, its effect along with the frequency limitations can be considered. Thus, a two-pole model for $R_{mi}(s)$ is given as:

$$R_{mi}(s) = \frac{R_{m0}}{\left(1 + \frac{s}{\omega_{p1}}\right) \left(1 + \frac{s}{\omega_{p2}}\right)} \quad (13)$$

Where ω_{p1} and ω_{p2} are the angular frequencies of the first and second pole and R_{m0} is the DC gain of the OTRA. For middle frequency applications, (13) can be expressed as:

$$R_{mi}(s) \approx \frac{1}{sC_{mi} \left(1 + \frac{s}{\omega_{p2}}\right)}, \quad C_{mi} = \frac{1}{R_{m0}\omega_{p1}} \quad (14)$$

For very high frequency applications, $R_{mi}(s)$ can be expressed as:

$$R_{mi}(s) \approx \frac{\omega_{p2}}{s^2 C_{mi}} \quad (15)$$

where C_{mi} is the parasitic capacitance associated to the first pole. Thus, if (15) is considered, (12) is modified as:

$$\frac{V_{Out}}{V_{in}} = \frac{\frac{C_m C_3}{C_1 C_2 \omega_{p2}} s^3 + \left(\frac{C_m}{C_1 C_2 \omega_{p2}} \left(\frac{1}{R_5} - \frac{1}{R_4} \right) + \frac{C_3}{C_2} \left(1 + \frac{C_m}{C_1} \right) \right) s^2 + \frac{s}{C_2} \left(\frac{1}{R_5} - \frac{1}{R_4} \right) \left(1 + \frac{C_m}{C_1} \right) + \frac{1}{R_1 R_3 C_1 C_2}}{\frac{C_m C_{m2}}{C_1 C_2 \omega_{p2}^2} s^4 + \frac{s^3}{\omega_{p2}} \left(\frac{C_m}{C_1} + \frac{C_{m2}}{C_2} + \frac{2C_m C_{m2}}{C_1 C_2} \right) + \left(1 + \frac{C_m}{C_1} + \frac{C_{m2}}{C_2} + \frac{C_m C_{m2}}{C_1 C_2} + \frac{C_m}{C_1 C_2 R_2 \omega_{p2}} \right) s^2 + \frac{1}{R_2 C_2} \left(1 + \frac{C_m}{C_1} \right) s + \frac{1}{R_1 R_6 C_1 C_2}} \quad (16)$$

For middle frequency applications:

$$\omega_{p1} \ll |s = j\omega| \ll \omega_{p2}, \quad R_{mi}(s) \approx \frac{1}{sC_{mi}} \quad (17)$$

Therefore, (12) can be approximated to:

$$\frac{V_{Out}}{V_{in}} = \frac{\frac{C_3}{C_2} \left(1 + \frac{C_m}{C_1} \right) s^2 + \frac{s}{C_2} \left(\frac{1}{R_5} - \frac{1}{R_4} \right) \left(1 + \frac{C_m}{C_1} \right) + \frac{1}{R_1 R_3 C_1 C_2}}{\left(1 + \frac{C_m}{C_1} + \frac{C_{m2}}{C_2} + \frac{C_m C_{m2}}{C_1 C_2} \right) s^2 + \frac{1}{R_2 C_2} \left(1 + \frac{C_m}{C_1} \right) s + \frac{1}{R_1 R_6 C_1 C_2}} \quad (18)$$

Therefore, ω_0 and Q are given by:

$$\omega_0 \approx \frac{1}{\sqrt{R_1 R_6 C_1 C_2 \left(1 + \frac{C_{m1}}{C_1} + \frac{C_{m2}}{C_2} + \frac{C_{m1} C_{m2}}{C_1 C_2}\right)}},$$

$$Q \approx \frac{R_2 C_1 C_2}{C_1 + C_{m1}} \sqrt{\frac{1 + \frac{C_{m1}}{C_1} + \frac{C_{m2}}{C_2} + \frac{C_{m1} C_{m2}}{C_1 C_2}}{R_1 R_6 C_1 C_2}}$$
(19)

For low frequency, C_{m1} , C_{m2} , ω_{p1} and ω_{p2} are insignificant, thus, ω_0 and Q can be approximated to:

$$\omega_0 \approx \frac{1}{\sqrt{R_1 R_6 C_1 C_2}}, \quad Q \approx R_2 \sqrt{\frac{C_2}{R_1 R_6 C_1}}$$
(20)

6.2. Non-inverting band-pass, inverting and non-inverting low-pass filters using ICCII±

As a second example, let us consider the non-inverting band-pass and low-pass filter using ICCII± as shown in **Fig. 19(a)** [24]. The independent voltage source is modeled according to **Fig. 4** and each active device is modeled as shown in **Fig. 8(b)** and **10(c)**, including their parasitic elements. Therefore, the equivalent circuit is illustrated in **Fig. 19(b)**. For this example, the formulation method introduced in Section 5 should be used. Thus, the nullators, norators, current sources, admittances and capacitances data-structures are given by **Table 1**.

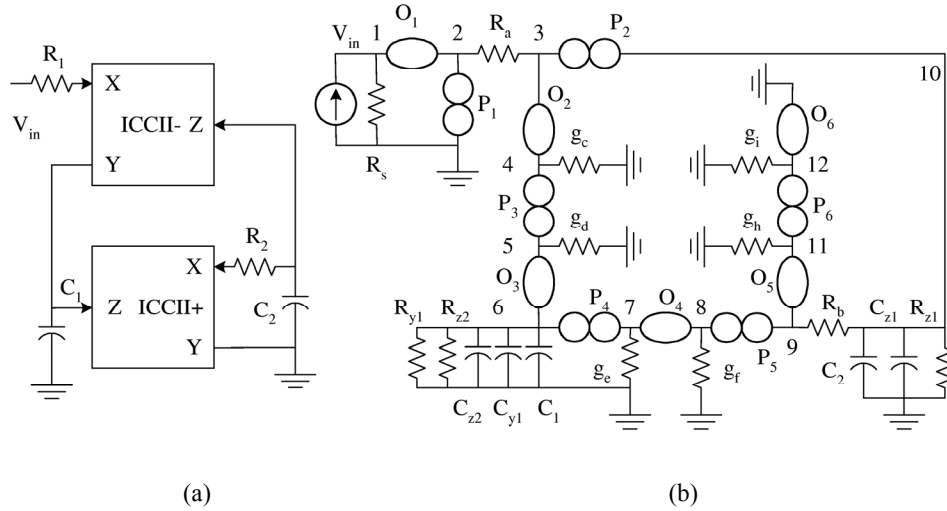


Figure 19: (a) ICCII±-based non-inverting band-pass and low-pass filter (b) nullor equivalent circuit.

Here:

$$g_a = \frac{1}{R_1 + R_{x1}}, \quad g_b = \frac{1}{R_2 + R_{x2}}, \quad C_a = C_1 + C_{z2} + C_{y1}, \quad C_b = C_2 + C_{z1}$$
(21)

The O and P vectors are calculated from **Table 1** and given as:

$$O = [\{1,2\}, \{3,4\}, \{5,6\}, \{7,8\}, \{9,11\}, \{10\}]$$

$$P = [\{1\}, \{3,10\}, \{4,5\}, \{6,7\}, \{8,9\}, \{11,12\}]$$
(22)

Therefore, the V vector is obtained directly from the O vector. The J vector is filled by comparing the nodes of the current source with each node of the P vector. Both vectors are given as:

$$V = [V_{\{1,2\}}, V_{\{3,4\}}, V_{\{5,6\}}, V_{\{7,8\}}, V_{\{9,11\}}, V_{\{10\}}]^T, \quad J = [V_{in}, 0, 0, 0, 0, 0]^T \quad (23)$$

Table 1: Data-Structures of Two-Terminal Elements

Nullators	Nodes	Norators	Nodes	Admittances	Nodes	Current Source	Nodes
O ₁	1,2	P ₁	2,0	g _s =1	1,0	V _{in}	0,1
O ₂	3,4	P ₂	3,10	g _a	2,3		
O ₃	5,6	P ₃	4,5	g _c =1	4,0		
O ₄	7,8	P ₄	6,7	g _d =1	5,0		
O ₅	9,11	P ₅	8,9	C _a	6,0		
O ₆	12,0	P ₆	11,12	g _{z2}	6,0		
				g _{y1}	6,0		
				g _e =1	7,0		
				g _f =1	8,0		
				g _b	9,10		
				g _h =1	11,0		
				g _i =1	12,0		
				g _{z1}	10,0		
				C _b	10,0		

Following the formulation method, the list $Col_{i(O_x)}$ are given as:

$$\begin{aligned}
 Col_{i(O_1)} &= [(1, g_s), (2, [3, g_a])] \\
 Col_{i(O_2)} &= [(3, [2, g_a]), (4, g_c)] \\
 Col_{i(O_3)} &= [(5, g_d), (6, g_{y1}, g_{z2}, C_a)] \\
 Col_{i(O_4)} &= [(7, g_e), (8, g_f)] \\
 Col_{i(O_5)} &= [(9, [10, g_b]), (11, g_h)] \\
 Col_{i(O_6)} &= [(10, [9, g_b]), g_{z1}, C_b]
 \end{aligned} \quad (24)$$

Finally, by using the P vector, the system of equations is given by:

$$J = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ -g_a & g_a & 0 & 0 & -g_b & -g_b + g_{z1} + sC_b \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & g_{z2} + g_{y1} + sC_a & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & g_b & -g_b \\ 0 & 0 & 0 & 0 & g_h & 0 \end{bmatrix} V \quad (25)$$

The non-inverting band-pass response is taken in the node 10. The inverting and non-inverting low-pass responses are available in the nodes 6 and 3, respectively. In this manner, by solving (25) to $V_{\{10\}}$, $V_{\{5,6\}}$ and $V_{\{3,4\}}$ the fully-symbolic transfer functions are computed as:

$$\frac{V_{\{10\}}}{V_{in}} = \frac{g_a(g_{y1} + g_{z2} + C_a s)}{C_a C_b s^2 + (C_a(g_b + g_{z1}) + C_b(g_{y1} + g_{z2})s + g_b(g_a + g_{y1} + g_{z2})) + g_{z1}(g_{y1} + g_{z2})} \quad (26)$$

$$\frac{V_{\{3,4\}}}{V_{in}} = -\frac{V_{\{5,6\}}}{V_{in}} = \frac{g_a g_b}{C_a C_b s^2 + (C_a(g_b + g_{z1}) + C_b(g_{y1} + g_{z2})s + g_b(g_a + g_{y1} + g_{z2})) + g_{z1}(g_{y1} + g_{z2})} \quad (27)$$

7. Conclusion

In this chapter, the generation of nullor-based models, not only of operational amplifiers but also of normal and inverting first, second and third generation current conveyors with a single and multiple outputs along with active devices with features that involves differential voltage signals has been introduced. From two-port and four-terminal network point of view, all the proposed models have been generated by taking into account the impedance levels associated to the input-output terminals along with the gain-equations of the active devices. As one can see throughout the chapter, the nullor-based models are not complex and they can quickly be included into symbolic analyzers. Further, nullor-based active device models by including parasitic elements, has also been introduced. Furthermore, a novel method to formulate the system of equations in order to compute fully-symbolic small-signal characteristics of analog circuits by applying only standard NA has been presented. Thus, by using the relationships of nullators and norators and by manipulating their data-structures, the admittance matrix can quickly be constructed, avoiding waste of CPU-time and memory in the formulation process. Examples have been introduced to show the usefulness of the nullor-based models and the potentiality of the proposed formulation method

Acknowledgements

The preparation of this chapter has been partially supported by Promep-Mexico under the project number: UATLX-PTC-088 and by Consejería de Innovación, Ciencia y Empresa, Junta de Andalucía under the project number: TIC-2532. The author thanks the support of the JAE-Doc program of CSIC, co-funded by FSE.

References

- [1] J. Vlach and K. Singhal, *Computer methods for circuit analysis and design*. Norwell, Massachusetts: Kluwer, 1993.
- [2] G. Gielen and W. Sansen, *Symbolic analysis for automated design of analog integrated circuits*. USA: Kluwer Academic Publishers, 1991.
- [3] F.V. Fernández, A. Rodríguez-Vázquez, J.L. Huertas, and G. Gielen, *Symbolic analysis techniques: applications to analog design automation*. Piscataway, NJ: IEEE Press, 1998.
- [4] H.J. Carlin, "Singular networks elements", *IEEE Transactions on Circuit Theory*, vol. 11, pp. 67-72, March 1964.
- [5] B.D.H. Tellegen, "On nullators and norators", *IEEE Transactions on Circuit Theory*, vol. CT-13, pp. 466-469, December 1966.

- [6] J.A. Svoboda, "Using nullors to analyse linear networks", *International Journal of Circuit Theory and Applications*, vol. 14, pp. 169–180, 1986.
- [7] J. A. Svoboda, "Current conveyors, operational amplifiers and nullors", *IEE Proceedings G Circuits, Devices & Systems*, vol. 136, pp. 317–322, December 1989.
- [8] A. Carlosena and G. S. Moschytz, "Nullators and norators in voltage to current mode transformations", *International Journal of Circuit Theory and Applications*, vol. 21, pp. 421–424, 1993.
- [9] H. Floberg, *Symbolic analysis in analog integrated circuit design*. USA: Kluwer Academic Publishers, 1997.
- [10] H. Schmid, "Approximating the universal active element", *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 47, pp. 1160–1169, November 2000.
- [11] C. Sánchez-López and E. Tlelo-Cuautle, "Behavioral model generation for symbolic analysis of analog integrated circuits", in *IEEE International Symposium on Signals, Circuits and Systems*, 2005, pp. 327–330.
- [12] C. Sánchez-López and E. Tlelo-Cuautle, "Behavioral model generation of current-mode analog circuits", in *IEEE International Symposium on Circuits and Systems*, 2009, pp. 2761–2764.
- [13] E. Tlelo-Cuautle, C. Sánchez-López, and D. Moro-Frías, "Symbolic analysis of (MO)(I)CCI(II)(III)-based analog circuits", *International Journal of Circuit Theory and Applications*, [Online] Doi:10.1002/cta.582, 2009.
- [14] A. M. Soliman and R. A. Saad, "On the voltage mirrors and the current mirrors", *Analog Integrated Circuits and Signal Processing*, vol. 32, pp. 79–81, July 2002.
- [15] E. Tlelo-Cuautle, C. Sánchez-López, E. Martínez-Romero, S. X.-D. Tan, "Symbolic analysis of analog circuits containing voltage mirrors and current mirrors", *Analog Integrated Circuits and Signal Processing*, [Online] DOI: 10.1007/s10470-010-9455-y, February 2010.
- [16] D. Biolk, R. Senani, V. Biolka, and Z. Kolka, "Active elements for analog signal processing: classification, review, and new proposals", *RadioEngineering*, vol. 17, pp. 15-32, December 2008.
- [17] A. Payne and C. Toumazou, "Analog amplifiers: classification and generalization", *IEEE Transactions on Circuits and Systems I*, vol. 43, pp. 43–50, January 1996.
- [18] J.H. Huijsing, "Operational floating amplifier", *IEE Proceedings G Circuits, Devices and Systems*, vol. 137, pp. 131–136, April 1990.
- [19] K.C. Smith and A. Sedra, "The current conveyor—A new circuit building block", *Proceedings of the IEEE*, vol. 56, pp. 1368–1369, August 1968.
- [20] A.S. Sedra and K. C. Smith, "A second generation current conveyor and its applications", *IEEE Transactions on Circuit Theory*, vol. CT-17, pp. 132–134, February 1970.
- [21] A.S. Sedra, G.W. Roberts, and F. Gohh, "The current conveyor: history, progress and new results", *IEE Proceedings G Circuits, Devices & Systems*, vol. 137, pp. 78–87, April 1990.
- [22] A. Fabre, "Third-generation current conveyor: A new helpful active element", *Electronics Letters*, vol. 31, pp. 338–339, March 1995.
- [23] B. Wilson, "Recent developments in current conveyors and current-mode circuits", *IEE Proceedings G Circuits, Devices & Systems*, vol. 137, pp. 63-77, April 1990
- [24] A.M. Soliman, "The inverting second generation current conveyors as universal building blocks", *International Journal of Electronics and Communications*, vol. 62, pp. 114-121, March 2008.
- [25] H.O. Elwan and A.M. Soliman, "Novel CMOS differential voltage current conveyor and its applications", *IEE Proceedings Circuits, Devices and Systems*, vol. 144, pp. 195–200, June 1997.
- [26] C. Sánchez-López, D. Moro-Frías, and E. Tlelo-Cuautle, "Improving the formulation process of the system of equations of analog circuits," in *International Workshop on Symbolic and Numerical Methods, Modeling and Applications to Circuit Design*, 2008, pp. 102–106.
- [27] E. Tlelo-Cuautle, E. Martínez-Romero, C. Sánchez-López, and S. X.-D. Tan, "Symbolic formulation method for mixed-mode analog circuits using nullors," in *IEEE International Conference on Electronics, Circuits and Systems*, 2009, pp. 856–859.
- [28] C. Shi and S. Tan, "Canonical symbolic analysis of large analog circuits with determinant decision diagrams", *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, vol. 19, pp. 1–18, January 2000.
- [29] S. Tan, "Symbolic analysis of analog circuits by boolean logic operations", *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 53, pp. 1313–1317, November 2006.
- [30] K.N. Salama and A.M. Soliman, "Active RC applications of the operational transresistance amplifier", *Frequenz*, vol. 54, pp. 171-176, January 2000.

Generation of the Transfer Functions for MIMO Systems

Mihai Iordache and Lucia Dumitriu*

Electrical Engineering Department, Politehnica University of Bucharest, Romania

Abstract: The main aim of the symbolic analysis is the generation of the symbolic expressions of the circuit functions and computation of their sensitivities, in order to evaluate the circuit characteristics and their variation in respect of the parameter values. Other important information for circuit designer is about the location of poles and zeros in the complex frequency plane, or about the eigenvalues of the state matrix, necessary to develop a qualitative analysis of the circuit. For multiple inputs (multiple excitations) and multiple outputs it is of interest to obtain the transfer functions involving the input-output variables. In this case, the concepts of controllability and observability are of importance.

In this chapter two approaches for transfer function generation both in Single-Input-Single-Output (SISO) and in Multiple-Input-Multiple-Output (MIMO) systems are presented: one approach is based on the state equations and the other one on the semi-state equations of the circuit. Two variants for the first approach and three variants for the second one are developed and compared on illustrative examples, pointing out the advantages.

Keywords: Symbolic analysis, transfer functions, matrix transfer functions, MIMO systems, design of analog circuits, state variables, state equations, normal tree, nodal analysis, modified nodal equations, frequency domain analysis, time-domain analysis.

1. Introduction

Optimization techniques together with the automatic and interactive procedures have a great impact on the computational effort needed in the symbolic circuit analysis and allow increasing the analog circuit size we can handle.

The methods used for the automatic generation in symbolic or numeric-symbolic form of the circuit functions (transfer impedance, driving-point impedance, transfer admittance, driving-point admittance, voltage gain and current gain) can be grouped in some classes: the determinant method [1-10], the signal – flow graph method [6], [11-15], the tree-enumeration method, [16-24], the parameter-extraction method [25], [26], the interpolation method [27-29], and the state variable method [12, 30-36].

Starting from the circuit function definition we shall develop two techniques for automatic generation of the transfer functions in symbolic or numeric-symbolic matrix form: one is based on state equations, and the other one - on semi-state equations [37, 38]. Both techniques allow the analysis of the circuits without any constraint regarding the type of circuit elements. Excess elements are also considered. For the linear passive circuits with zero initial conditions and one input port and one output port, the general definition of the circuit function (also called network or transfer function) is:

$$F_{oi}(s) = H(s) = \frac{\mathcal{L}(m_o)}{\mathcal{L}(m_i)} = \frac{M_o(s)}{M_i(s)} \quad (1)$$

*Address correspondence to Lucia Dumitriu: Electrical Engineering Department, Politehnica University of Bucharest, Romania; E-mail: lucia.dumitriu@gmail.com.

where $\mathcal{L}(m_o)=M_o(s)$ and $\mathcal{L}(m_i)=M_i(s)$ represent the Laplace transform of the output variable, and of the input signal, respectively. Taking into account the possible nature of m_o and m_i , four types of transfer functions can be defined: transfer impedance, transfer admittance, voltage gain, and current gain. According with the notations in **Fig. 1**, the definitions of the four transfer functions are:

$$Z_{oi} = \left. \frac{d V_o(s)}{J_i(s)} \right|_{I_o=0}, \quad Y_{oi} = \left. \frac{d I_o(s)}{E_i(s)} \right|_{V_o=0}, \quad A_{oi} = \left. \frac{d V_o(s)}{E_i(s)} \right|_{I_o=0} \quad \text{and} \quad B_{oi} = \left. \frac{d I_o(s)}{J_i(s)} \right|_{V_o=0} \quad (2)$$

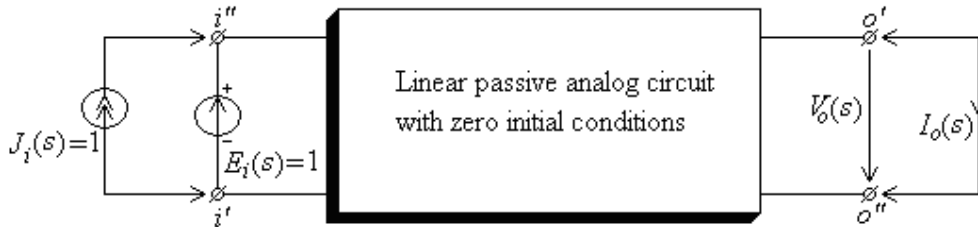


Figure 1: The input/output of the two-port equivalent scheme for the network function definitions.

In the case of an impulse signal $\delta(t)$, $\mathcal{L}\{\delta(t)\} = 1$, the transfer function coincides with the output variable.

In order to define the driving-point impedance (admittance), the input/output ports are treated as in **Fig. 2(a, b)**.

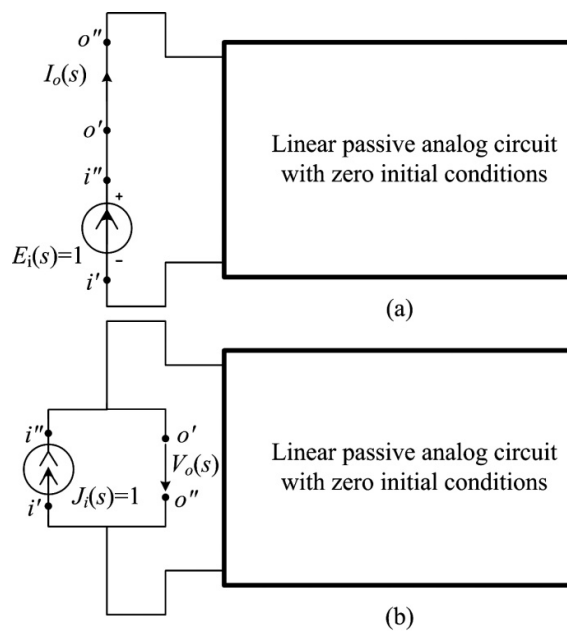


Figure 2: The input/output of the two-port equivalent scheme for the definitions of the driving-point impedance (a) and admittance (b).

The linear and time-invariant circuits under consideration can contain: all four types of linear controlled sources, resistors, inductors, capacitors, nullors (for ideal opamps operating in the linear mode), and any multiterminal or multiport circuit element having an equivalent scheme made up only by two-terminal elements and controlled

sources. The circuits can also have Multiple Inputs And Multiple Outputs (MIMO) [39]. Consider a linear circuit C , with p_i input ports and p_o output ports. The p_i input ports consist in p_{vi} voltage ports, characterized by the voltages $v_k = e_k$, $k = 1, \dots, p_{vi}$, and p_{ci} current ports, characterized by the currents $i_k = j_k$, $k = 1, \dots, p_{ci}$ ($p_i = p_{vi} + p_{ci}$). In order to identify the output voltage (current) ports we can consider them as ideal independent current (voltage) sources having the currents (voltages) equal to zero. The transfer function matrix $\mathbf{H}(s)$ has $(p_o \times p_i)$ dimension. The rows of the $\mathbf{H}(s)$ matrix correspond to the output variables and the columns to the input (excitation) signals.

Remarks

Computation of a transfer function in respect of a certain input signal implies that all the other input signals are zero.

2. State Variable Approach

2.1. Introduction

The state variable approach is very attractive both for numeric and symbolic analysis [37, 39, 42], because of the following advantages:

- The knowledge of the state variable values at time t_0 , and of the excitations applied at $t \geq t_0$, is enough to determine the state of the system at any time $t > t_0$;
- After finding the state variable vector $\mathbf{x}(t)$, we can easily find the time domain response of the circuit both in numeric or symbolic form;
- Finding the poles and zeros of a transfer function, in numeric or numeric-symbolic form, for a linear circuit (or for a piecewise-linear circuit in the operating point), is reduced to finding eigenvalues of two real matrices;
- The state variable approach can be extended without much difficulty to deal with nonlinear circuits in which the Laplace transform method does not apply;
- The state space representation provides a convenient and compact way to model and analyze multiple-inputs multiple outputs systems either in numeric or in symbolic form.

There are two methods to formulate the state equations of a circuit in normal-form:

1. The hybrid method [12];
2. The topological method of the normal tree [12, 30, 33, 34, 40].

The first method involves a very large number of calculations and for this reason it is not used to set up the state equations in normal form.

In this section the topological method of the normal tree, based on Kirchhoff's laws and on the constitutive equations of the circuit elements, is developed. A general class of time-invariant analog circuits, including circuits with excess elements, can be handled by this procedure.

We call circuits with excess elements of the first kind (with degenerations of the first kind) the circuits that contain [30, 34]:

- Loops of capacitors (C -loops), and/or loops of capacitors and independent voltage sources ($C-E_i$ loops), and/or loops of capacitors and controlled voltage sources ($C-E_c$ loops) and/or loops of capacitors, independent voltage sources, and controlled voltage sources ($C-E_i-E_c$ loops) – generic called $C-E$ loops;
- Cutsets of inductors (L cutsets), and/or cutsets of inductors and independent current sources ($L-J_i$ cutsets), and/or cutsets of inductors and controlled current sources ($L-J_c$ cutsets), and/or cutsets of inductors, independent current sources, and controlled current sources ($L-J_i-J_c$ cutsets) – generic called $L-J$ cutsets.

The system of linearly independent first-order differential equations obtained by processing the Kirchhoff's laws and the constitutive equations in the form,

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, t) \quad (3)$$

is called *state equation in normal form* of the circuit. In (3) \mathbf{x} is a set of n independent variables $\mathbf{x} = [x_1, x_2, \dots, x_n]^t$ called *state vector*, whose components x_1, x_2, \dots, x_n are *state variables*, and n is the *complexity order* of the circuit.

One of the important problems in the state variable approach is the selection of state variables. Usually the inductor currents and the capacitor voltages are chosen as state variables. It is generally accepted that the number of state variables associated with a circuit is equal to the number of dynamic elements minus the number of excess elements of the first kind, [30, 34], so that the complexity order of the circuit is given by:

$$n = n_L + n_C - n_{L-E} - n_{\Sigma L-J} \quad (4)$$

If the state equation in normal-form exists, it can be expressed in the following form:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \quad (5)$$

for the circuits without excess elements, and also for the circuits with degenerations that do not contain E_i and/or J_i . The occurrence of such degenerations involves the following normal form of the state equation:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} + \mathbf{B}_1\dot{\mathbf{u}} \quad (6)$$

In the above equations \mathbf{u} is the input vector of m size (independent sources), \mathbf{x} is the state vector of n size, \mathbf{A} , \mathbf{B} , and \mathbf{B}_1 are real matrices of appropriate dimensions. It is important to note that:

1. \mathbf{A} , called the state matrix of the circuit, is always a square matrix of order n ; its eigenvalues are *all the natural frequencies of the circuit* so their knowledge allows a complete qualitative analysis: stability and nature of the circuit response;
2. \mathbf{B}_1 exists only if the independent sources connected to the input ports make with the circuit elements degenerations of the first kind.

The loops of inductors and/or loops of inductors and independent and/or controlled voltage sources ($L-E$ loops), and the cutsets of capacitors and/or cutsets of capacitors and independent and/or controlled current sources ($C-J$ cutsets) are degenerates of the second kind and they decide dependencies among capacitor derivative voltages and inductor derivative currents. Each $L-E$ loop (each $C-J$ cutset) introduces an eigenvalue equal to zero.

We have to point up some advantages of the state variable method:

- The state equations in symbolic form improve the accuracy in the numerical calculations;
- The order of a circuit is equal to the rank of the associated state matrix \mathbf{A} in (5) and (6). For a non-singular matrix \mathbf{A} , the order of the circuit is equal to the number of eigenvalues of \mathbf{A} , or the number of natural frequencies of the circuit;
- Knowing the eigenvalues of the state matrix allows a qualitative analysis even for the nonobservable and noncontrolable circuits (for this kind of circuits the set of the transfer function poles are included in the set of the state matrix eigenvalues, but are not coincident, because some poles disappear in the computing process);

- The natural frequencies of a circuit directly affect the stability behavior of the circuit. For (asymptotic) stability, all the natural frequencies must be in the left half of the complex s plane;
- In the case of the linear time-invariant analog circuits, the state matrix in symbolic form can be further used to obtain a simplified expression of any circuit function;
- It's easy to estimate an optimal time step in relation with the eigenvalues of the state-matrix. So, the smallest eigenvalue (largest time constant) allows estimating a total simulation time, and the largest eigenvalue (smallest time constant) facilitates the choice of the time step size;

2.2. Assumptions on the class of analyzed circuits

We assume that the analyzed circuits meet the following requirements [30]:

1. Consistency assumptions

- a) There are not loops consisting only of independent and/or controlled voltage sources, called E loops;
- b) There are not cutsets made up only of independent and/or controlled current sources, called J cutsets;
- c) All nonlinear capacitors are voltage-controlled. That means that the charge q of each capacitor is a function of its voltage, $q = \tilde{q}(v)$;
- d) All nonlinear inductors are current-controlled. That means that the flux φ of each inductor is a function of its current, $\varphi = \hat{\varphi}(i)$;

2. Normal tree assumptions

- a) A special tree called *Normal Tree (NT)* is chosen. The normal tree has to contain in this priority: all independent and controlled voltage sources, all nonlinear voltage-controlled resistors, as many capacitors as possible, as many controlling branches of the current-controlled voltage sources and of the current-controlled current sources as possible (these branches are considered as resistive branches having the resistances equal to zero), and as many resistors as possible. The normal tree will not contain independent or controlled current sources, and nonlinear current-controlled resistors;
- b) Any controlled source belonging to a C - E loop or to an L - J cutset can depend only on the voltage of a tree capacitor or on the current of a cotree inductor.

Though most practical circuits do not violate these assumptions, regarding the last one, we can note that there are many circuits which, while do not meet it, they still have state equations in normal-form [30].

In fact, even in those rare cases when a circuit violates one or more of the above assumptions, we could easily overcome the difficulty by introducing small parasites to the circuit. For example, a small resistance can be inserted in series with a capacitor or a voltage source belonging to a C - E loop to remove it. Similarly, a small conductance can be connected in parallel with an inductor or a current source belonging to an L - J cutset to eliminate it.

2.3. Setting-up of the state equations in symbolic/numeric normal form

Consider, in the general case, a nonlinear network containing both linear and nonlinear resistors, inductors, and capacitors, independent voltage and current sources, linear magnetic couplings, the four types of linear controlled sources, and any multiterminal circuit element having an equivalent scheme made up only of two-terminal circuit elements and controlled sources. The magnetic couplings can be simulated by inductors and current derivative-controlled voltage sources [33, 34].

According with the above assumptions a normal tree is selected.

The capacitors that are not included in the normal tree are called *excess capacitors*, and the inductors that are included in the normal tree are called *excess inductors*.

A cutset associated to a tree-branch is a cutset made up of only one tree branch and many links, whose direction agrees with the tree branch direction.

We built the fundamental cutset matrix \mathbf{D} with the following rules:

- arrange the links in the order: C_l (link capacitors), R_l (link resistors), L_l (link inductors), J_c (controlled current sources), J_i (independent current sources);
- arrange the cutsets associated to the tree-branches in the order: $\Sigma(E_i)$ (independent voltage sources), $\Sigma(E_c)$ (controlled voltage sources), $\Sigma(C_t)$ (tree capacitors), $\Sigma(R_t)$ (tree resistors), L_t (tree inductors).

We obtain [33, 34, 43, 44]:

$$\mathbf{D} = \begin{matrix} \Sigma/l & C_l & R_l & L_l & J_c & J_i \\ \Sigma(E_i) & \mathbf{D}_{E_i C} & \mathbf{D}_{E_i R} & \mathbf{D}_{E_i L} & \mathbf{D}_{E_i J_c} & \mathbf{D}_{E_i J_i} \\ \Sigma(E_c) & \mathbf{D}_{E_c C} & \mathbf{D}_{E_c R} & \mathbf{D}_{E_c L} & \mathbf{D}_{E_c J_c} & \mathbf{D}_{E_c J_i} \\ \Sigma(C_t) & \mathbf{D}_{CC} & \mathbf{D}_{CR} & \mathbf{D}_{CL} & \mathbf{D}_{CJ_c} & \mathbf{D}_{CJ_i} \\ \Sigma(R_t) & \mathbf{0} & \mathbf{D}_{RR} & \mathbf{D}_{RL} & \mathbf{D}_{RJ_c} & \mathbf{D}_{RJ_i} \\ \Sigma(L_t) & \mathbf{0} & \mathbf{0} & \mathbf{D}_{LL} & \mathbf{D}_{LJ_c} & \mathbf{D}_{LJ_i} \end{matrix} \quad (7)$$

where, for example, E_i (J_c) is the set of independent voltage (controlled current) sources, and \mathbf{D}_{CC} (\mathbf{D}_{LL}) represents the incidence submatrix of the link capacitors (link inductors) to the cutsets associated to the tree branch capacitors (inductors). The null submatrices \mathbf{D}_{RC} , \mathbf{D}_{LC} , \mathbf{D}_{LR} in (7) appear as a consequence of the above definition of the normal tree.

With Kirchhoff's laws we express the tree-branch currents in respect of the link currents, and the link voltages in respect of the tree-branch voltages as it follows:

$$\mathbf{i}_t = -\mathbf{D}\mathbf{i}_l \quad (8)$$

$$\mathbf{v}_l = \mathbf{D}^t \mathbf{v}_t \quad (9)$$

Adding to the equation system (8), (9) the constitutive equations of the linear and of the nonlinear circuit elements (the nonlinear characteristics will be approximated by piecewise-linear continuous curves) a complete equation system will be obtained.

Choosing as state variables the tree capacitor voltages \mathbf{v}_{C_t} and the cotree inductor currents \mathbf{i}_{L_t} , the above system is solved by a program that implements the algorithm presented in [33, 36], and uses for manipulation of the symbolic expressions, the symbolic simulator Maple. The result of the computing process will be the state equations in symbolic/numeric normal (5) or (6) where:

$$\mathbf{x} = \begin{bmatrix} \mathbf{v}_{C_t}^t & \mathbf{i}_{L_t}^t \end{bmatrix}^t, \quad \mathbf{u} = \begin{bmatrix} \mathbf{v}_{E_i}^t & \mathbf{i}_{J_i}^t \end{bmatrix}^t \quad (10)$$

are the state vector, and the vector of the input signals, respectively.

The output vector \mathbf{y} has the following expression:

$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u} \quad (11)$$

Algorithm for symbolic generation of the state equations in normal form

- Step 1.** A normal tree and its corresponding cotree are selected;
- Step 2.** Expressing the currents of the current-controlled sources in respect of their controlling variables (according to the definition relations), and taking into account that $i_C = C \cdot dv_C/dt$, the KCL is applied on the cutsets associated to the tree branch capacitors;
- Step 3.** Expressing the voltages of the voltage-controlled sources in respect of their controlling variables, and taking into account that: $v_R = R \cdot i_R$, $v_L = L \cdot di_L/dt$, the KVL is applied on the loops associated to the link capacitors;
- Step 4.** Apply the KVL on the loops associated to the link inductors;
- Step 5.** Apply the KCL on the cutsets associated to the tree branch inductors;
- Step 6.** Write the KCL on the cutsets associated to the tree branch resistors;
- Step 7.** Write the KVL on the loops associated to the link resistors;
- Step 8.** Considering as independent variables the current vectors i_{Rt} and i_{Rl} , the equations from step 6 and 7 are solved;
- Step 9.** The symbolic/numeric expressions obtained at the step 8 are introduced into the equations from the step 3 and 5. The equations generated in this way, together with the expressions from the step 8 are introduced into the equations from the steps 2 and 4. Solving these equations in respect of the differentials of the state variables, the state equations in normal symbolic/numeric form are obtained.

Example 1

Let us consider the nonreciprocal electric circuit with excess elements shown in **Fig. 3**. The normal tree branches are drawn by bold lines, $NT = \{e_1, C_3, R_2, L_5\}$. This circuit has an excess element of the first kind (an L - J cutset, $\Sigma_{L-J} = \{L_5, L_6, j_7\}$), and an excess element of the second kind (an L - E loop, $l_{L-E} = \{e_1, L_5, L_6\}$). This loop determines an eigenvalue equal to zero. The complexity order of the circuit is $n = n_L + n_C - n_{\Sigma_{L-J}} = 2 + 1 - 1 = 2$.

Hence, the structure of the state vector and of the input vector is:

$$\mathbf{x} = [v_{C3}, i_{L6}]^t, \mathbf{u} = [j_7] \tag{12}$$

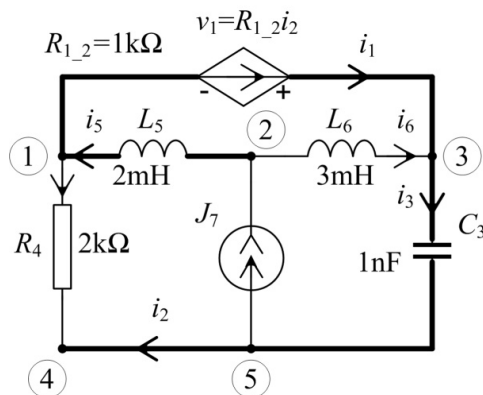


Figure 3: Circuit diagram.

Step 1. A normal tree and its corresponding cotree are selected: $NT = \{e_1, C_3, R_2, L_5\}$ and $CT = \{R_4, L_6, j_7\}$.

Step 2. Applying the KCL on the cutset associated to the tree capacitor C_3 , we obtain:

$$(\Sigma_{C_3}): C_3 \frac{dv_{C_3}}{dt} = i_4 + j_7 \quad (13)$$

Step 3. The circuit has not link capacitors.

Step 4. The KVL applied on the loop associated to the link inductor L_6 has the following structure:

$$(l_{L_6}): L_6 \frac{di_{L_6}}{dt} = L_5 \frac{di_{L_5}}{dt} - R_{1_2} i_2 \quad (14)$$

Step 5. The KCL applied on the cutset associated to the tree branch inductor L_5 has the form:

$$(\Sigma_{L_5}): i_{L_5} = -i_{L_6} + j_7 \quad (15)$$

Step 6. The KCL applied on the cutset associated to the tree branch resistor ($R_2 = 0$, that identifies the controlling current of e_1) has the following structure:

$$(\Sigma_{R_2}): i_2 = i_4 \quad (16)$$

Step 7. The KVL applied on the loop associated to the link resistor R_4 has the form:

$$(l_{R_4}): R_4 i_4 = -v_{C_3} + R_{1_2} i_2 \quad (17)$$

Step 8. Solving the equations from steps 6 and 7, we get the current of the tree branch resistor i_2 , and the link resistor current i_4 :

$$i_2 = i_4 = -\frac{1}{R_4 - R_{1_2}} v_{C_3} \quad (18)$$

Step 9. The state equation in normal form is:

$$\frac{d}{dt} \begin{bmatrix} v_{C_3} \\ i_{L_6} \end{bmatrix} = \begin{bmatrix} -\frac{1}{C_3(R_4 - R_{1_2})} & 0 \\ \frac{R_{1_2}}{(L_5 + L_6)(R_4 - R_{1_2})} & 0 \end{bmatrix} \begin{bmatrix} v_{C_3} \\ i_{L_6} \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} [j_7] + \begin{bmatrix} 0 \\ \frac{L_5}{L_5 + L_6} \end{bmatrix} \frac{d}{dt} [j_7] \quad (19)$$

The state matrix in (19) has a zero column; consequently it has a null eigenvalue. The cause is the second kind degeneration $l_{L-E} = \{e_1, L_5, L_6\}$.

Example 2

Let us consider the nonlinear circuit in **Fig. 4**. This is a circuit without excess elements, which contains two nonlinear resistors. The voltage-controlled resistor is represented by a parallel equivalent scheme (**Fig. 16(a)**), while the current-controlled resistor is modeled with a series equivalent scheme (**Fig. 16(b)**).

The complexity order of the circuit is $n_{Ct} + n_{Ll} = 2 + 2 = 4$, and the state variables are v_{C2} , v_{C3} , i_{L4} , and i_{L5} .

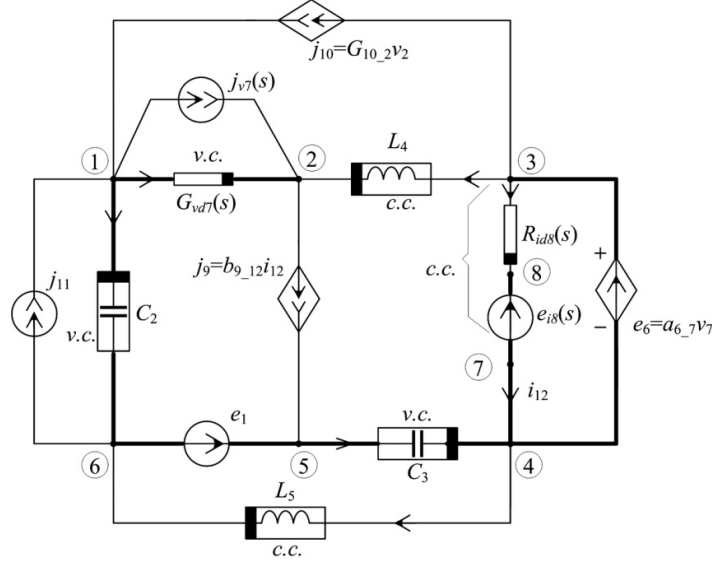


Figure 4: Circuit without excess elements.

We can remark that one of the controlled sources (j_{10}) has as controlling variable a state variable of the circuit (v_{C2}). To formulate the state equations in symbolic form, the normal tree $\{e_1, e_6, e_{i8}, C_2, C_3, b_{12}, G_{vd7}\}$ is automatically generated and the symbolic state equations are obtained:

$$\begin{aligned}
 \frac{dv_{C2}}{dt} &= \frac{G_{10_2}}{C_{d2}(v_{C2})} v_{C2} + \frac{G_{vd7}(s) \cdot R_{id8}(s)}{C_{d2}(v_{C2}) [G_{vd7}(s) \cdot R_{id8}(s) - a_{6_7} b_{9_12}]} i_{L4} + \\
 &+ \frac{1}{C_{d2}(v_{C2})} j_{11} + \frac{a_{6_7} b_{9_12}}{C_{d2}(v_{C2}) [G_{vd7}(s) \cdot R_{id8}(s) - a_{6_7} b_{9_12}]} j_{v7}(s) + \\
 &+ \frac{b_{9_12} G_{vd7}(s)}{C_{d2}(v_{C2}) [G_{vd7}(s) \cdot R_{id8}(s) - a_{6_7} b_{9_12}]} e_{i8}(s) \\
 \frac{dv_{C3}}{dt} &= \frac{G_{10_2}}{C_{d3}(v_{C3})} v_{C2} + \frac{1}{C_{d3}(v_{C3})} i_{L4} + \frac{1}{C_{d3}(v_{C3})} i_{L5} \\
 \frac{di_{L4}}{dt} &= -\frac{1}{L_{d4}(i_{L4})} v_{C2} - \frac{1}{L_{d4}(i_{L4})} v_{C3} - \frac{(1 + a_{6_7}) R_{id8}(s)}{L_{d4}(i_{L4}) [G_{vd7}(s) \cdot R_{id8}(s) - a_{6_7} b_{9_12}]} i_{L4} - \\
 &- \frac{1}{L_{d4}(i_{L4})} e_1 - \frac{(1 + a_{6_7}) R_{id8}(s)}{L_{d4}(i_{L4}) [G_{vd7}(s) \cdot R_{id8}(s) - a_{6_7} b_{9_12}]} j_{v7}(s) - \\
 &- \frac{b_{9_12} (1 + a_{6_7})}{L_{d4}(i_{L4}) [G_{vd7}(s) \cdot R_{id8}(s) - a_{6_7} b_{9_12}]} e_{i8}(s) \\
 \frac{di_{L5}}{dt} &= -\frac{1}{L_{d5}(i_{L5})} v_{C3} + \frac{1}{L_{d5}(i_{L5})} e_1
 \end{aligned} \tag{20}$$

Example 3

The circuit in **Fig. 5** contains an excess element (an L - J cutset), so that the complexity order of the circuit is $n_c + n_L - n_{\Sigma L-J} = 1 + 2 - 1 = 2$. The controlling variables of the controlled sources are variables associated to resistors.

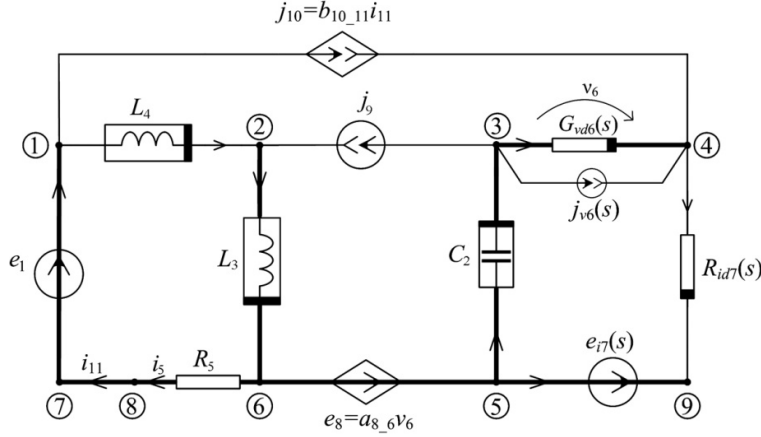


Figure 5: Circuit with an excess element.

According to the assumptions presented above, the program generates the normal tree introducing the inductor L_3 in the tree branch set:

$NT = \{e_1, e_8, e_{i7}, C_2, b_{11}, R_5, G_{vd6}, L_3\}$. The state variables of the circuit are the voltage of the tree branch capacitor, and the current of the link inductor, $\{v_{C2}, i_{L4}\}$. The symbolic state equations obtained by the program have the form:

$$\begin{aligned} \frac{dv_{C2}}{dt} = & -\frac{G_{vd6}(s)}{[1+G_{vd6}(s) \cdot R_{id7}(s)]C_{d2}(v_{C2})}v_{C2} + \frac{b_{10-11}G_{vd6}(s) \cdot R_{id7}(s)}{(-1+b_{10-11})[1+G_{vd6}(s) \cdot R_{id7}(s)]C_{d2}(v_{C2})}i_{L4} + \\ & + \frac{1}{C_{d2}(v_{C2})}j_9 + \frac{1}{[1+G_{vd6}(s) \cdot R_{id7}(s)]C_{d2}(v_{C2})}j_{v6}(s) - \frac{G_{vd6}(s)}{[1+G_{vd6}(s) \cdot R_{id7}(s)]C_{d2}(v_{C2})}e_{i7}(s) \end{aligned} \quad (21)$$

$$\frac{di_{L4}}{dt} = \frac{R_5}{(-1+b_{10-11})[L_{d3}(i_3)+L_{d4}(i_{L4})]}i_{L4} + \frac{1}{L_{d3}(i_3)+L_{d4}(i_{L4})}e_1 - \frac{L_{d3}(i_3)}{L_{d3}(i_3)+L_{d4}(i_{L4})} \frac{dj_9}{dt}$$

Example 4

In **Fig. 6** a nonreciprocal circuit with magnetic couplings is represented. The branches of the normal tree are drawn by bold lines – $NT = \{e_1, C_2, L_3, L_4, R_7, b_8, C_{10}, R_{11}, R_{12}, e_{13}\}$. This circuit has two excess inductors – L_3 and L_4 (there are two L cut sets), so the structure of the state vector and the input vector is:

$$\mathbf{x} = [v_{C2}, v_{C10}, i_{L5}, i_{L6}]^t, \mathbf{u} = [e_{13}] \quad (22)$$

The state equations in symbolic normal form are:

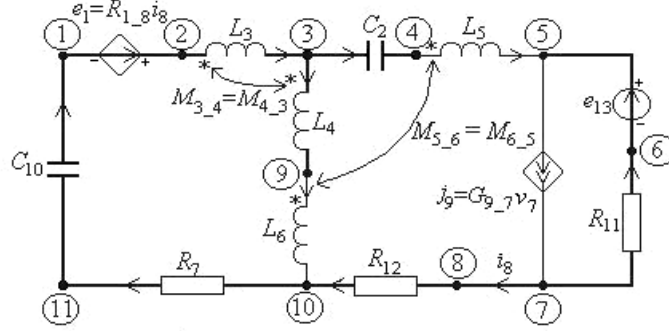


Figure 6: Circuit with magnetic couplings.

$$\frac{dv_{C2}}{dt} = \frac{1}{C_2} i_{L5}$$

$$\frac{di_{L5}}{dt} = \frac{L_3 + L_4 + L_6 + M_{3_4} + M_{4_3}}{\alpha} v_{C2} + \frac{L_4 + L_6 - M_{5_6} + M_{4_3}}{\alpha} v_{C10} + \frac{\beta}{\alpha} i_{L5} + \frac{\gamma}{\alpha} i_{L6} + \frac{L_3 + L_4 + L_6 + M_{3_4} + M_{4_3}}{\alpha} e_{13}$$

$$\frac{di_{L6}}{dt} = -\frac{L_3 + M_{4_3} + M_{6_5}}{\alpha} v_{C2} - \frac{-L_5 + M_{4_3} + M_{6_5}}{\alpha} v_{C10} - \frac{\chi}{\alpha} i_{L5} - \frac{\delta}{\alpha} i_{L6} - \frac{L_3 + M_{4_3} + M_{6_5}}{\alpha} e_{13}$$

where:

$$\alpha = M_{4_3} M_{5_6} + M_{4_3} M_{5_6} + L_3 M_{5_6} + M_{5_6} M_{6_5} + L_3 M_{6_5} + M_{6_5} M_{3_4} - L_5 L_6 - L_4 L_5 - L_5 M_{4_3} - L_3 L_5 - L_5 M_{3_4} - L_3 L_6 - L_3 L_4$$

$$\beta = -L_3 R_{11} G_{9_7} R_7 - M_{4_3} R_{11} G_{9_7} R_7 + L_3 R_{12} - M_{4_3} R_{1_8} + L_3 R_{11} + L_6 R_{11} + L_6 R_7 + L_6 R_{12} + L_4 R_{11} + L_4 R_7 + L_4 R_{12} + M_{3_4} R_{11} + M_{3_4} R_{12} + M_{5_6} R_{1_8} + M_{4_3} R_{12} + M_{4_3} R_7 + M_{4_3} R_{11} - L_6 R_{1_8} - L_4 R_{1_8} - M_{5_6} R_7 - L_6 R_{11} G_{9_7} R_7 - M_{3_4} R_{11} G_{9_7} R_7 - L_4 R_{11} G_{9_7} R_7$$

$$\gamma = R_{11} G_{9_7} R_7 (-L_3 - M_{4_3} - M_{3_4} - L_4 - L_6) + R_7 (L_4 - M_{5_6} + M_{4_3} + L_6) \quad (24)$$

$$\chi = -M_{6_5} R_{1_8} - L_5 R_7 + L_3 R_{12} + M_{6_5} R_{11} - M_{4_3} R_{11} G_{9_7} R_7 - L_3 R_{11} G_{9_7} R_7 - M_{6_5} R_{11} G_{9_7} R_7 + M_{4_3} R_{11} + L_3 R_{11} + M_{4_3} R_{12} + M_{6_5} R_7 + M_{4_3} R_7 - M_{4_3} R_{1_8} + L_5 R_{1_8} + M_{6_5} R_{12}$$

$$\delta = R_{11} G_{9_7} R_7 (-L_3 - M_{3_4} - M_{6_5}) + R_7 (-L_5 + M_{4_3} + M_{6_5})$$

Example 5

The essential quality of the state variable approach is its ability to yield all the natural frequencies of the circuit, which are the eigenvalues of the state matrix A . A good example is the $\mu A741$ operational amplifier [45], shown in **Fig. 7**, whose (partially) symbolic state equations in normal form will be formulated, in order to extract the circuit state matrix. If the transistors are modeled as in **Fig. 8**, the small-signal equivalent circuit of the amplifier in open-loop configuration contains 26 nodes and 140 primitive elements.

The circuit was analyzed with SYSEG [34] by using a diakoptic approach [36] in the following steps:

Step 1. Applying the decomposition technique.

Step 2. Setting up the state equations in symbolic form of each subcircuit.

Step 3. Aggregation of these equations to obtain the state equations in normal form of the entire circuit.

In the case of the large-scale circuits, the full symbolic expressions being too large, the symbolic manipulator (in our case Maple) fails. This is why we solved for the equations of the entire circuit in partially-symbolic form, taking as symbols different parameters. The result covers 9 pages in 9 pt font. By extracting the state matrix of (24x24) size, in partially-symbolic form, and using appropriate approximation methods, we can get the dominant eigenvalues of the circuit. These are very important in computer aided design for stability analysis.

The numeric computation of the 24 eigenvalues gives:

Eigenvalues:=[-41982289128.32411, -30212173475.16874, -15457407820.87884, -15865105030.38149, -10345252310.39025, -8441468555.58094, -2632995640.68324, -202065956.40661, -2158354872.80714+696171274.30908*I, -6476416.47260, -2158354872.80714-696171274.30908*I, -1256476779.39262, -962243603.16744, -875147990.25975, -626648837.11704, -385273800.27889, -41478072.27004 + 87013622.64242*I, -41478072.27004 - 87013622.64242*I, -235102888.68789, -255322896.51432, -204756738.00733, -45060097.86413, **-2067.32751, -19.50929**].

We retain the dominant eigenvalues **-19.50929** and **-2067.32751** the closest to the complex plane origin (**Fig. 9**). If the simulation is performed by SPICE simulator (that uses modified nodal analysis) the voltage gain is a rational function whose denominator is a polynomial of 23-th order. The poles of the transfer function have the same numerical values as the above eigenvalues except the second dominant value that was lost in the computation process by a pole-zero cancellation (actually the two values are closed). Consequently, excepting the case when the generated transfer function is in irreducible form, the complete information about the natural frequencies of a circuit can be obtained only by state variable approach. In other words the set of the transfer function poles is included in the set of the state matrix eigenvalues $\{p\} \subseteq \{\lambda\}$.

Remarks

1. Natural (called also critical or characteristic) frequencies are circuit parameters which dictate the characteristic dynamic behavior of the circuit, independently of the excitation at the input ports. They directly affect the stability behavior of the circuit.

2. *Controllability* provides that an input is available to bring the initial state to any desired state in a finite time. A network is completely controllable if all its natural frequencies are coupled to all the inputs of the network.
3. A natural frequency that cannot be excited from an input, it is said to be non-controllable at this input. *Non-controllable natural frequencies may cause a state variable to be unstable, independently of the input excitation.*
4. *Observability* involves a relation between an output variable and the state vector, so that knowing an output trajectory provides enough information to predict the initial state of the system. A network is completely observable if all its natural frequencies are coupled to all the outputs of the network.
5. A natural frequency uncoupled to an output is said to be non-observable at that output. If such a parameter exists, *the network may seem stable at that output, although it is internally unstable.*
6. A completely controllable and completely observable network has the transfer function in irreducible form. Only for these networks the set of the poles is coincident with the set of the natural frequencies.
7. Pole-zero cancellation is theoretically allowed (the expressions are equivalent) but practically dangerous because one lost information (there might be a signal that cannot be observed).
8. Assume the pole that was cancelled out *is unstable*; if the system is not completely observable *one cannot see the instability at the output. One of the natural frequencies doesn't appear in the output expression, but it appears explicitly in one of the states (the system is not internally stable).*
9. It is not detectable if the pole-zero cancellation affects an unstable pole, that is why one should not cancel out any pole-zero pair of the transfer function before testing for stability.
10. Because the eigenvalues of the state matrix are all the natural frequencies of the circuit, a pole-zero cancellation have to be analyzed from a state space perspective.

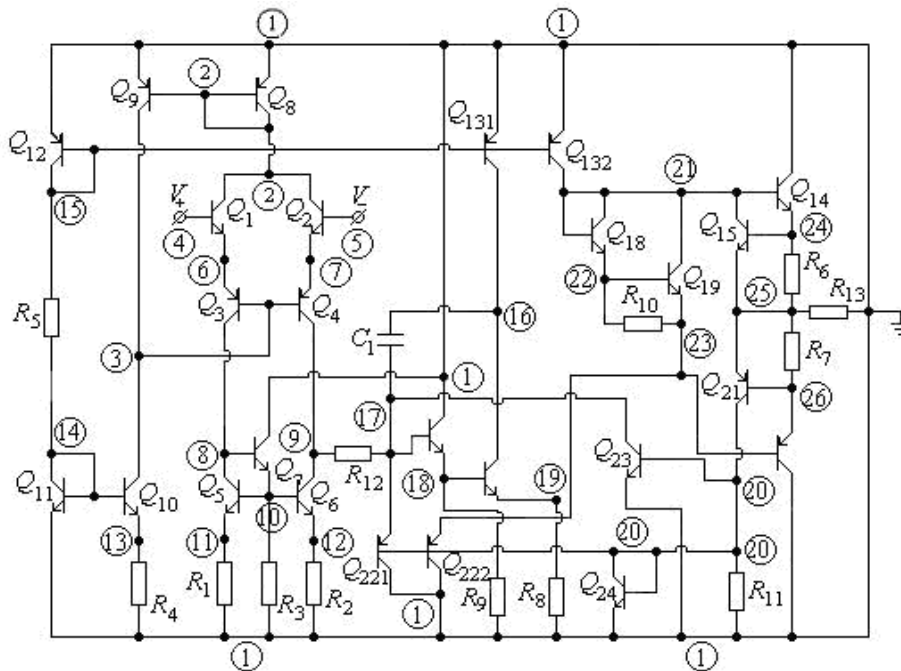


Figure 7: The μA741 operational amplifier.

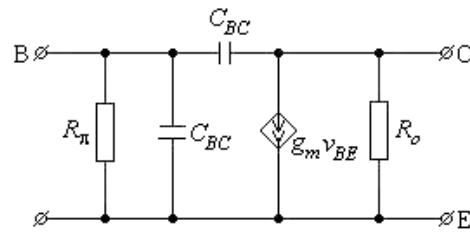


Figure 8: AC model for bipolar transistors.

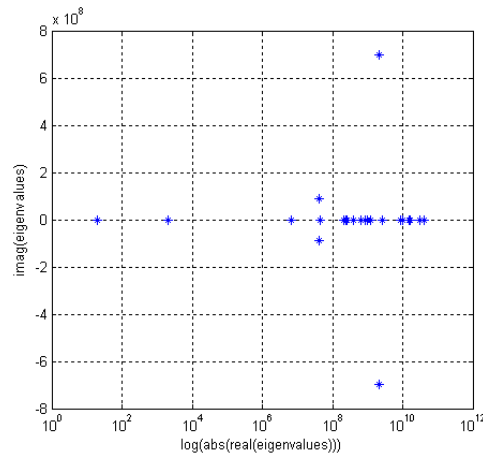


Figure 9: Eigenvalue locations of $\mu\text{A}741$ operational amplifier.

2.4. Transfer function generation for MIMO systems

2.4.1. Transfer function generation in matrix form

The state equations in frequency domain (with null initial conditions) and the corresponding output equations, obtained by applying the Laplace transform to (6) and (11), have the following form:

$$\begin{aligned} (s\mathbf{I} - \mathbf{A})\mathbf{X}(s) &= (\mathbf{B} + s\mathbf{B}_1)\mathbf{U}(s) \\ \mathbf{Y}(s) &= \mathbf{C}\mathbf{X}(s) + \mathbf{D}\mathbf{U}(s) \end{aligned} \quad (25)$$

where the state vector $\mathbf{X}(s)$ is an n -vector of Laplace transforms of the state variables, the input vector $\mathbf{U}(s)$ is an m -vector of Laplace transforms of the input signals and the output vector $\mathbf{Y}(s)$ is a p -vector of Laplace transforms of the output variables. The matrices \mathbf{A} , \mathbf{B} , \mathbf{B}_1 , \mathbf{C} and \mathbf{D} , which are specific to the circuit, have $n \times n$, $n \times m$, $n \times m$, $p \times n$ and $p \times m$ dimension respectively, and \mathbf{I} is the unit matrix. Starting from the definition, the transfer function $\mathbf{H}(s)$, in matrix form, $p \times m$, has the expression

$$\mathbf{H}(s) = \frac{\mathbf{Y}(s)}{\mathbf{U}(s)} = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}(\mathbf{B} + s\mathbf{B}_1) + \mathbf{D} \quad (26)$$

The rows of the $\mathbf{H}(s)$ matrix correspond to the output variables and the columns to the input (excitation) signals. For example, the entry $H_{kj}(s)$ corresponds to the output variable from the port k (V_k or I_k) and to the input signal (E_j or J_j) from the port j .

The algorithm for automatic transfer function generation according with (26) consists in the following:

Step 1. Setting up the state equations in time domain (6) by automatic generation of the matrices A , B , B_1 ;

Step 2. Automatic generation of the matrices C and D ;

Step 3. Transform the state equations from the time domain in frequency domain (25), by applying Laplace transform;

Step 4. Inverse the matrix $(sI-A)$ that contains as symbol the complex frequency s ;

Step 5. Compute the transfer function matrix $H(s)$ by using (26).

Remarks

If the matrix that has to be inverted at the step 4 is too large, and/or when we want to keep also certain circuit parameters as symbols (for example in order to compute the transfer function sensitivities) it is more efficient to generate the transfer functions for MIMO systems according to the procedure presented further.

2.4.2. Transfer function generation without matrix inversion

The algorithm that avoids matrix inversion, suitable for symbolic analysis, involves the following steps:

Step 1: Starting from the input file of the netlist type, the state equations in time domain (6) are generated;

Step 2: Considering null initial conditions, the state equations in time domain are transformed in the frequency domain. These equations have a partial symbolic form containing as symbols the complex frequency, all the input signals, and also some circuit parameters, specified by the user;

Step 3: The partial symbolic algebraic equations obtained in the previous step are solved in respect of the Laplace transforms of the state variables by a suitable program (for example Maple). In the case of the large-scale analog circuits we can use a reduction algorithm of the state equation number [41].

Step 4: The output variables are expressed in respect of the inputs and of the complex variable s using the second equation in (25);

Step 5: Taking into account the definitions (2), the network functions are performed. Obviously, for the computation of a transfer function in respect of an input signal, all the other signals in the corresponding equation have to be zero.

Example 6

In **Fig. 10** a universal filter is presented. Let us compute the transfer functions of the four filters.

The output variable vector is

$$Y(s) = [V_{4_11}(s) \quad V_{6_11}(s) \quad V_{8_11}(s) \quad V_{10_11}(s)]^t \quad (27)$$

and the input signal

$$U(s) = [E_{1_11}(s)] \quad (28)$$

The vector of the four transfer functions is:

$$H(s) = [A_{4,11_1,11} \quad A_{6,11_1,11} \quad A_{8,11_1,11} \quad A_{10,11_1,11}]^t \quad (29)$$

Computing the matrix of the transfer functions by using both of the above procedures we get the same expression $H(s) = TF_ST_EQ$:

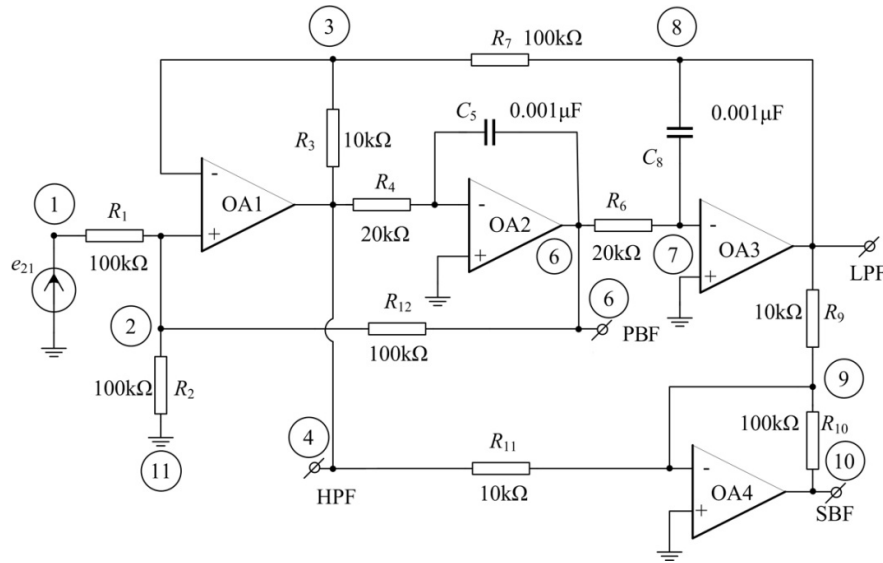
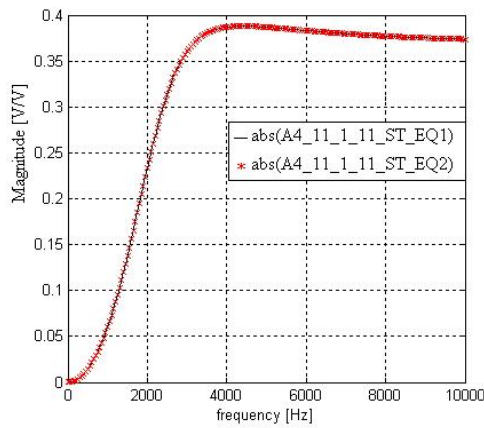


Figure 10: Universal filter diagram.

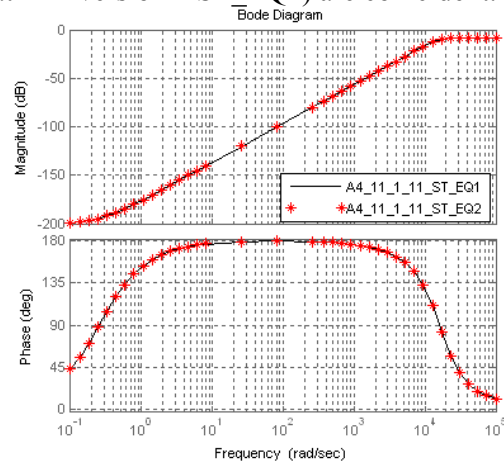
$$TF_{ST_EQ} := \frac{\frac{24980. + 0.3667 s^2 + 1.611 s}{s^2 + 18330. s + 0.2499 10^9} \cdot \frac{0.7332 10^7 s + 18330.}{400. s^2 + 0.7332 10^7 s + 0.9996 10^{11}} \cdot \frac{0.3666 10^{12}}{400. s^2 + 0.7332 10^7 s + 0.9996 10^{11}}}{\frac{3.666 s^2 - 2.220 s + 0.9165 10^{10}}{s^2 + 18330. s + 0.2499 10^9}}$$

In Figs. 11 and 12, the frequency characteristics of the voltage gain magnitude for the High Pass Filter (HPF), and for the Band Pass Filter (BPF), respectively, are presented.

The two curves obtained by the two procedures based on the state equations (with matrix inversion – ST_EQ1 and without matrix inversion – ST_EQ2) are coincident.



(a)



(b)

Figure 11: Frequency characteristics of the voltage gain for HPF (a), and Bode characteristics of the voltage gain for HPF (b), by the state variable approach.

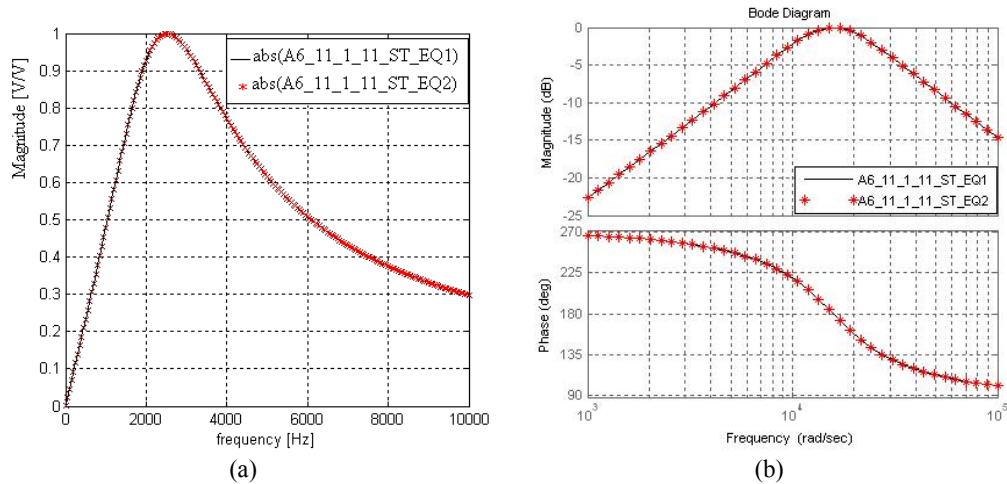


Figure 12: Frequency characteristics of the voltage gain for BPF (a), and Bode characteristics of the voltage gain for BPF (b), by the state variable approach.

3. Semi - State Variable Approach

3.1. Introduction

The nodal approach in circuit analysis has two variants:

- The first - using companion circuits (resistive discrete circuits associated with an integration implicit algorithm), that gets the dynamic circuit response by the analysis of a sequence of resistive circuits. In the Modified Nodal Analysis (MNA) the circuit matrix is obtained by augmentation of the nodal conductance (admittance) matrix corresponding to the non-NA-compatible circuit elements with additional rows and columns.
- The second - using the circuit equations in dynamic behavior obtained by KCL and the constitutive equations of the circuit elements. The dynamic behavior description of the circuit by modified nodal equations is known as the semi-state variable method, and uses those variables which are most convenient from the point of view of the analysis results.

The independent variables of both methods above are:

- $n - 1$ node voltages, that make up the voltage vector v_{n-1} ;
- m branch currents, which can not be expressed in respect to the node voltages or to the first order derivatives of the node voltages and the circuit parameters, that make up the vector of controlling currents i_m ;

The vector i_m contains the currents of the independent voltage sources, the currents of voltage sources controlled both in current or in voltage, the controlling currents of the Current-Controlled Voltage Sources (CCVSs) and of the Current-Controlled Current Sources (CCCSs), the currents of the current-controlled nonlinear resistors, the currents of the current-controlled nonlinear inductors, and the linear inductor (magnetic coupled or not) currents.

In dynamic behavior, the inductor (capacitor) equations contain current (voltage) derivatives. Consequently, the modified nodal equations must contain derivatives of the independent variables. Their general form is:

$$\begin{cases} W\dot{x}(t) + Gx(t) = Bu(t) \\ y(t) = L'x(t) \end{cases} \quad (30)$$

where:

$$\mathbf{x}(t) = \begin{bmatrix} \mathbf{v}_{n-1} \\ \mathbf{i}_m \end{bmatrix} \quad (31)$$

is the circuit independent variable vector, with initial condition $\mathbf{x}_0 = \mathbf{x}(t_0)$;

$$\dot{\mathbf{x}}(t) = \frac{d}{dt} \begin{bmatrix} \mathbf{v}_{n-1} \\ \mathbf{i}_m \end{bmatrix} \quad (32)$$

is the derivative of the independent variable vector;
 \mathbf{W} and \mathbf{G} are square matrices $(n-1+m) \times (n-1+m)$;

$$\mathbf{u} = \begin{bmatrix} \mathbf{i}_{sc,n-1} \\ \mathbf{e}_m \end{bmatrix} \quad (33)$$

is the input vector, with $\mathbf{i}_{sc,n-1}$ - the vector of the short-circuit currents, and \mathbf{e}_m is the electromotive force (e.m.f.) vector of the ideal independent voltage sources, and of the sources resulted from the nonlinear characteristics of the nonlinear current-controlled elements which are piece-wise linearized.

\mathbf{B} and \mathbf{L} are selector matrices, with entries $(-1, 0$ or $1)$, and the superscript “t” denotes the transpose.

In frequency domain and with zero initial conditions, (30) can be written as:

$$\begin{cases} (\mathbf{G} + s\mathbf{W})\mathbf{X}(s) = \mathbf{B}\mathbf{U}(s) \\ \mathbf{Y}(s) = \mathbf{L}^t \mathbf{X}(s) \end{cases} \quad (34)$$

For the case of nonlinear elements, the modified nodal equations of the circuit become:

$$\begin{cases} \mathbf{W}(\mathbf{x}(t)) \cdot \dot{\mathbf{x}}(t) + \mathbf{G}\mathbf{x}(t) + \mathbf{F}(\mathbf{x}(t)) = \mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) = \mathbf{L}^t \mathbf{x}(t) \end{cases} \quad (35)$$

where $\mathbf{F}(\mathbf{x}(t))$ is a nonlinear function of \mathbf{x} .

If for the nonlinear inductors (nonlinear capacitors) the magnetic fluxes (electrical charges) are considered as independent variables, the matrix \mathbf{W} is independent of \mathbf{x} .

Remarks

The method can be applied to a large class of nonlinear circuits containing both linear and nonlinear non-monotonic resistors, inductors and capacitors, independent voltage and current sources, the four types of linear controlled sources, time-variable resistors (as models for the switches) and any type of excess elements. It has a lot of advantages:

- the semi-state equations are simply formulated from the netlist;
- it does not impose restrictions to the nonlinear characteristics;
- it does not require an associated discrete equivalent circuit which needs an additional computing effort, especially when the numeric integration is made with changeable time step;
- the matrices \mathbf{W} and \mathbf{G} are sparse, consequently, they require just a little space of memory in a computer implementation; this compensates the disadvantage to operate with a large number of variables;

- by simply cancellation of the matrix W and by solving an algebraic system of equations, the DC analysis can be performed; this can be necessary for finding initial conditions of the dynamic behaviors;
- the method can be applied for nonlinear and/or switching circuits;
- for the nonlinear circuit analysis, the method has the great advantage that allows keeping as symbols only the parameters of the nonlinear elements; so the elements of W and G matrices, depending on the linear circuit elements, are computed only once, at the beginning of the iteration procedure.

3.2. Contributions of several ideal circuit elements to the modified nodal equations in time-domain

Linear inductor

The constitutive equation in connection with the notations in **Fig. 13** is:

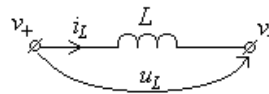


Figure 13: Linear inductor.

$$u_L = v_+ - v_- = \frac{d\phi_L}{dt} = L \frac{di_L}{dt} \Rightarrow -L \frac{di_L}{dt} + v_+ - v_- = 0 \tag{36}$$

The linear inductor introduces as new independent variable the current i_L . The contribution of this circuit element to the semi-state (time domain modified nodal) equations is:

$$\begin{bmatrix} \dot{v}_+ & \dot{v}_- & \dot{i}_L \\ \bullet & \bullet & 0 \\ \bullet & \bullet & 0 \\ 0 & 0 & -L \end{bmatrix} \cdot \begin{bmatrix} \dot{v}_+ \\ \dot{v}_- \\ \dot{i}_L \end{bmatrix} + \begin{bmatrix} v_+ & v_- & i_L \\ \bullet & \bullet & +1 \\ \bullet & \bullet & -1 \\ +1 & -1 & 0 \end{bmatrix} \cdot \begin{bmatrix} v_+ \\ v_- \\ i_L \end{bmatrix} = \begin{bmatrix} \bullet \\ \bullet \\ 0 \end{bmatrix} \tag{37}$$

Linear capacitor

The characteristic equation, using the notations in **Fig. 14**, is:

The two curves obtained by the two procedures based on the state equations are coincident.

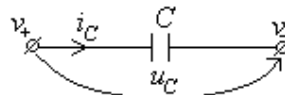


Figure 14: Linear capacitor.

$$i_C = \frac{dq_C}{dt} = C \frac{du_C}{dt} = C \frac{dv_+}{dt} - C \frac{dv_-}{dt} \tag{38}$$

According with (38), the contribution of a linear capacitor to the time domain modified nodal equations is:

$$\begin{bmatrix} \dot{v}_+ & \dot{v}_- \\ C & -C \\ -C & C \end{bmatrix} \cdot \begin{bmatrix} \dot{v}_+ \\ \dot{v}_- \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} v_+ \\ v_- \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad (39)$$

Magnetic coupled inductors

The constitutive equations, in connection with the notations in **Fig. 15**, are:

$$\begin{aligned} u_{L1} = v_{1+} - v_{1-} &= L_1 \frac{di_{L1}}{dt} + L_{12} \frac{di_{L2}}{dt} \Rightarrow -L_1 \frac{di_{L1}}{dt} - L_{12} \frac{di_{L2}}{dt} + v_{1+} - v_{1-} = 0 \\ u_{L2} = v_{2+} - v_{2-} &= L_2 \frac{di_{L2}}{dt} + L_{21} \frac{di_{L1}}{dt} \Rightarrow -L_2 \frac{di_{L2}}{dt} - L_{21} \frac{di_{L1}}{dt} + v_{2+} - v_{2-} = 0 \end{aligned} \quad (40)$$

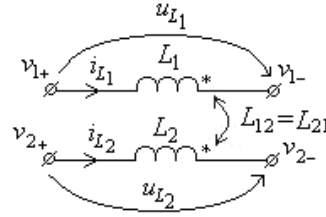


Figure 15: Magnetic coupled inductors.

The magnetic coupled linear inductors introduce as new independent variables the currents i_{L1} and i_{L2} . The contributions of these circuit elements to the semi-state (time domain modified nodal) equations are:

$$\begin{bmatrix} \dot{v}_{1+} & \dot{v}_{1-} & \dot{v}_{2+} & \dot{v}_{2-} & \dot{i}_{L1} & \dot{i}_{L2} \\ \bullet & \bullet & \bullet & \bullet & 0 & 0 \\ \bullet & \bullet & \bullet & \bullet & 0 & 0 \\ \bullet & \bullet & \bullet & \bullet & 0 & 0 \\ \bullet & \bullet & \bullet & \bullet & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & -L_1 & -L_{12} \\ 0 & 0 & 0 & 0 & -L_{21} & -L_2 \end{bmatrix} \cdot \begin{bmatrix} \dot{v}_{1+} \\ \dot{v}_{1-} \\ \dot{v}_{2+} \\ \dot{v}_{2-} \\ \dot{i}_{L1} \\ \dot{i}_{L2} \end{bmatrix} + \begin{bmatrix} v_{1+} & v_{1-} & v_{2+} & v_{2-} & i_{L1} & i_{L2} \\ \bullet & \bullet & \bullet & \bullet & +1 & 0 \\ \bullet & \bullet & \bullet & \bullet & -1 & 0 \\ \bullet & \bullet & \bullet & \bullet & 0 & +1 \\ \bullet & \bullet & \bullet & \bullet & 0 & -1 \\ \hline +1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & +1 & -1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} v_{1+} \\ v_{1-} \\ v_{2+} \\ v_{2-} \\ i_{L1} \\ i_{L2} \end{bmatrix} = \begin{bmatrix} \bullet \\ \bullet \\ \bullet \\ \bullet \\ 0 \\ 0 \end{bmatrix} \quad (41)$$

Voltage-controlled nonlinear resistor

Approximating the characteristics of all voltage-controlled nonlinear resistors by piecewise-linear continuous curves, for an arbitrary segment k , we can write:

$$i_{Rv} = \hat{i}_{Rv}(v_{Rv}) = G_{dRv}(k) \cdot v_{Rv} + j_{Rv}(k) \Leftrightarrow i_{Rv} = G_{dRv}(k) \cdot (v_{Rv}^+ - v_{Rv}^-) + j_{Rv}(k) \quad (42)$$

According with (42), the contribution of a voltage-controlled nonlinear resistor to the time-domain modified nodal equations is:

$$\begin{bmatrix} \dot{v}_{Rv}^+ & \dot{v}_{Rv}^- \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \dot{v}_{Rv}^+ \\ \dot{v}_{Rv}^- \end{bmatrix} + \begin{bmatrix} G_{dRv}(k) & -G_{dRv}(k) \\ -G_{dRv}(k) & G_{dRv}(k) \end{bmatrix} \cdot \begin{bmatrix} v_{Rv}^+ \\ v_{Rv}^- \end{bmatrix} = \begin{bmatrix} -j_{Rv}(k) \\ j_{Rv}(k) \end{bmatrix} \quad (43)$$

Current-controlled nonlinear resistor

Approximating the characteristics of all current-controlled nonlinear resistors by piecewise-linear continuous curves, for an arbitrary segment k , we can write:

$$v_{Ri} = \hat{v}_{Ri}(i_{Ri}) = R_{dRi}(k) \cdot i_{Ri} + e_{Ri}(k) \Rightarrow v_{Ri}^+ - v_{Ri}^- - R_{dRi}(k) \cdot i_{Ri} = e_{Ri}(k) \quad (44)$$

The current-controlled nonlinear resistor introduces as new independent variable the current i_{Ri} . Taking into account (44), the contribution of these circuit elements to the time-domain modified nodal equations can be expressed in matrix form as follows:

$$\begin{array}{ccc|ccc} \dot{v}_{Ri}^+ & \dot{v}_{Ri}^- & \dot{i}_{Ri} & v_{Ri}^+ & v_{Ri}^- & i_{Ri} \\ \left[\begin{array}{ccc|ccc} \bullet & \bullet & 0 \\ \bullet & \bullet & 0 \\ \hline 0 & 0 & 0 \end{array} \right] \cdot \begin{bmatrix} \dot{v}_{Ri}^+ \\ \dot{v}_{Ri}^- \\ \dot{i}_{Ri} \end{bmatrix} + \left[\begin{array}{ccc|ccc} \bullet & \bullet & & +1 & & \\ \bullet & \bullet & & -1 & & \\ \hline +1 & -1 & & -R_{dRi}(k) & & \end{array} \right] \cdot \begin{bmatrix} v_{Ri}^+ \\ v_{Ri}^- \\ i_{Ri} \end{bmatrix} = \begin{bmatrix} \bullet \\ \bullet \\ e_{Ri}(k) \end{bmatrix} \end{array} \quad (45)$$

According with the above equations, each voltage-controlled (current-controlled) nonlinear resistor can be substituted, for the arbitrary segment k , by the equivalent circuit in **Fig. 16(a, b)**.

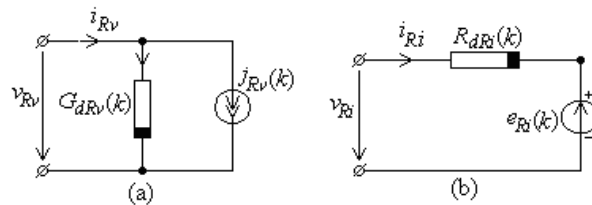


Figure 16: Equivalent circuits for piecewise-linear resistors: (a) –voltage-controlled resistor; (b) – current-controlled resistor.

Ideal independent voltage source

The constitutive equation of an ideal independent voltage source (**Fig. 17**) is:

$$u_E = v_+ - v_- = -e \quad (46)$$

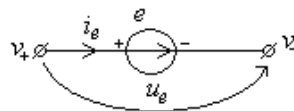


Figure 17: Ideal independent voltage source.

The source current i_E can not be expressed in respect of the node voltages v_+ , v_- , and the e.m.f. e . For this reason it becomes independent variable.

Taking into account (46), the contribution of this circuit element to the semi-state equations can be expressed in matrix form as:

$$\begin{array}{ccc|ccc} \dot{v}_+ & \dot{v}_- & \dot{i}_E & v_+ & v_- & i_E \\ \left[\begin{array}{ccc|ccc} \bullet & \bullet & 0 \\ \bullet & \bullet & 0 \\ \hline 0 & 0 & 0 \end{array} \right] \cdot \begin{bmatrix} \dot{v}_+ \\ \dot{v}_- \\ \dot{i}_E \end{bmatrix} + \left[\begin{array}{ccc|ccc} \bullet & \bullet & & +1 & & \\ \bullet & \bullet & & -1 & & \\ \hline +1 & -1 & & 0 & & \end{array} \right] \cdot \begin{bmatrix} v_+ \\ v_- \\ i_E \end{bmatrix} = \begin{bmatrix} \bullet \\ \bullet \\ -e \end{bmatrix} \end{array} \quad (47)$$

Ideal independent current source

The characteristic equation of the ideal independent current source, using the notations in **Fig. 18**, is:

$$i_J = j \tag{48}$$

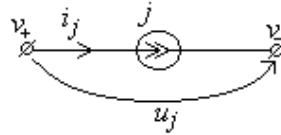


Figure 18: Ideal independent current source.

The contribution of this circuit element to the semi-state equations is:

$$\begin{bmatrix} \dot{v}_+ & \dot{v}_- \\ \bullet & \bullet \end{bmatrix} \cdot \begin{bmatrix} \dot{v}_+ \\ \dot{v}_- \end{bmatrix} + \begin{bmatrix} \bullet & \bullet \\ \bullet & \bullet \end{bmatrix} \cdot \begin{bmatrix} v_+ \\ v_- \end{bmatrix} = \begin{bmatrix} -j \\ +j \end{bmatrix} \tag{49}$$

Current-controlled voltage source

According with the notations in **Fig. 19**, the element equations are:

$$\begin{aligned} v_{C+} - v_{C-} &= 0 \\ v_{c+} - v_{c-} &= -R_{cC}i_C \quad \text{or} \quad v_{c+} - v_{c-} + R_{cC}i_C = 0 \end{aligned} \tag{50}$$

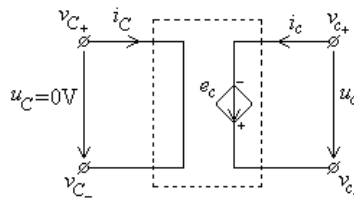


Figure 19: Current-controlled voltage source.

The variables of a current-controlled voltage source are v_{C+} , v_{C-} , v_{c+} , v_{c-} , i_C and i_c , and the contribution to the equation matrices is:

$$\begin{bmatrix} \bullet & \bullet & \bullet & \bullet & 0 & 0 \\ \bullet & \bullet & \bullet & \bullet & 0 & 0 \\ \bullet & \bullet & \bullet & \bullet & 0 & 0 \\ \bullet & \bullet & \bullet & \bullet & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \dot{v}_{C+} \\ \dot{v}_{C-} \\ \dot{v}_{c+} \\ \dot{v}_{c-} \\ \dot{i}_C \\ \dot{i}_c \end{bmatrix} + \begin{bmatrix} \bullet & \bullet & \bullet & \bullet & +1 & 0 \\ \bullet & \bullet & \bullet & \bullet & -1 & 0 \\ \bullet & \bullet & \bullet & \bullet & 0 & +1 \\ \bullet & \bullet & \bullet & \bullet & 0 & -1 \\ \hline +1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & +1 & -1 & R_{cC} & 0 \end{bmatrix} \cdot \begin{bmatrix} v_{C+} \\ v_{C-} \\ v_{c+} \\ v_{c-} \\ i_C \\ i_c \end{bmatrix} = \begin{bmatrix} \bullet \\ \bullet \\ \bullet \\ \bullet \\ 0 \\ 0 \end{bmatrix} \tag{51}$$

Voltage-controlled current source

The equations according to the notations in **Fig. 20** are:

$$\begin{aligned} i_C &= 0 \\ i_c &= G_{cC}u_C = G_{cC}v_{C+} - G_{cC}v_{C-} \end{aligned} \tag{52}$$

3.3. Transfer function generation in matrix form

3.3.1. Transfer function generation in matrix form by one matrix inversion

We recall (34)

$$\begin{cases} (\mathbf{G} + s\mathbf{W})\mathbf{X}(s) = \mathbf{B}U(s) \\ \mathbf{Y}(s) = \mathbf{L}^t \mathbf{X}(s) \end{cases}$$

and using the definition of the transfer function in matrix form, we get the following expression:

$$\mathbf{H}(s) = \frac{\mathbf{Y}(s)}{U(s)} = \mathbf{L}^t (\mathbf{G} + s\mathbf{W})^{-1} \mathbf{B} \quad (54)$$

3.3.2. Transfer function generation in matrix form by two matrices inversion

Multiplying both sides of the first equation in the system (34) by \mathbf{G}^{-1} (the matrix \mathbf{G} being a nonsingular matrix), we can write:

$$\begin{cases} (\mathbf{I} - s\mathbf{P})\mathbf{X}(s) = \mathbf{R}U(s) \\ \mathbf{Y}(s) = \mathbf{L}^t \mathbf{X}(s) \end{cases} \quad (55)$$

where

$$\mathbf{P} = -\mathbf{G}^{-1}\mathbf{W}, \quad \mathbf{R} = \mathbf{G}^{-1}\mathbf{B} \quad (56)$$

From (55), the network function $\mathbf{H}(s)$ takes the form:

$$\mathbf{H}(s) = \frac{\mathbf{Y}(s)}{U(s)} = \mathbf{L}^t (\mathbf{I} - s\mathbf{P})^{-1} \mathbf{R} \quad (57)$$

Because the expression (57) needs the inversion of two matrices, \mathbf{G} and $(\mathbf{I} - s\mathbf{P})$, the formula (54) is more advantageous.

3.3.3. Transfer function generation without matrix inversion

A more efficient procedure to generate network functions in matrix form, based on the semi-state equations in frequency domain (34), consists in the following steps:

Step 1: Starting from the input file of the netlist type (in which the circuit can contain multiple inputs) the semi-state equations in time-domain are generated;

Step 2: Considering null initial conditions, the semi-state equations in time domain are transformed in the frequency domain (using Laplace transform);

Step 3: The algebraic equations obtained in the previous step are solved in respect of the Laplace transforms of the state variables. In the case of large-scale analog circuits we can use the reduction algorithm of the semi-state equation number [41];

Step 4: The suitable output variables are expressed in respect of the inputs and of the complex variable s ;

Step 5: Taking into account the definitions, all network functions are performed.

Remarks

- For MIMO systems the most efficient procedure to generate the transfer functions is the procedure based on the direct solving, either of the state equations or of the semi-state equations, in the complex frequency domain, because these techniques

do not inverse any matrix. Because of this, they allow a large number of symbols that is, without doubt, a great advantage for the automatic design;

- In general, the dimension of $(sI - A)$ matrix is much smaller than the dimensions of $(sW + G)$ and $(I - sP)$ matrices, but it is less sparse.

Example 8

Applying the three above procedures (Transfer function matrix computed by the first semi-state procedure – TF_{ss1} (§ 3.3.1), Transfer function matrix computed by the second semi-state procedure – TF_{ss2} (§ 3.3.2), and Transfer function matrix computed by the third semi-state procedure – TF_{ss3} (§ 3.3.3)) we have computed the transfer functions of the universal filter in **Fig. 10**. Because of the huge full symbolic expressions of the transfer functions, the numerical values of the circuit parameters were substituted to get the following condensed expressions in partially numeric matrix form:

$$TF_{ss1} := \left[\begin{array}{c} \frac{0.001375 (51. + 404. s + 800. s^2)}{0.7500 10^9 + 55000. s + 3. s^2} \\ - \frac{137.5 (101. + 400. s)}{0.7500 10^9 + 55000. s + 3. s^2} \\ \frac{0.2750 10^{10}}{0.7500 10^9 + 55000. s + 3. s^2} \\ - \frac{0.05500 (0.5000 10^{12} + 101. s + 200. s^2)}{0.7500 10^9 + 55000. s + 3. s^2} \end{array} \right]$$

$$TF_{ss2} := \left[\begin{array}{c} - \frac{0.1000 10^{-24} (-0.4676 10^{24} - 0.3704 10^{25} s - 0.7334 10^{25} s^2 + 37. s^3)}{0.5000 10^9 + 36660. s + 2. s^2} \\ - \frac{0.1000 10^{-15} (400. s^2 + 0.3667 10^{21} s + 0.9260 10^{20})}{0.5000 10^9 + 36660. s + 2. s^2} \\ \frac{0.1000 10^{-21} (0.1000 10^{15} s^2 + 0.6508 10^{19} s + 0.1834 10^{32})}{0.5000 10^9 + 36660. s + 2. s^2} \\ - \frac{0.2000 10^{-8} (0.9165 10^{19} - 0.1917 10^{12} s + 0.3657 10^{10} s^2)}{0.5000 10^9 + 36660. s + 2. s^2} \end{array} \right]$$

$$TF_{ss3} := \left[\begin{array}{c} \frac{0.8800 10^{18} (s + 0.2550) (s + 0.2500)}{0.2400 10^{19} s^2 + 0.4399 10^{23} s + 0.5998 10^{27}} \\ - \frac{0.1111 10^{23} + 0.4400 10^{23} s}{0.2400 10^{19} s^2 + 0.4399 10^{23} s + 0.5998 10^{27}} \\ \frac{0.2200 10^{28}}{0.2400 10^{19} s^2 + 0.4399 10^{23} s + 0.5998 10^{27}} \\ - \frac{0.5500 10^{13} + 1111. s + 2200. s^2}{600. s^2 + 0.1100 10^8 s + 0.1500 10^{12}} \end{array} \right]$$

In **Figs. 21** and **22** the frequency characteristics of the voltage gain magnitude for the High Pass Filter (HPF), and for the Band Pass Filter (BPF), respectively, are presented for comparison.

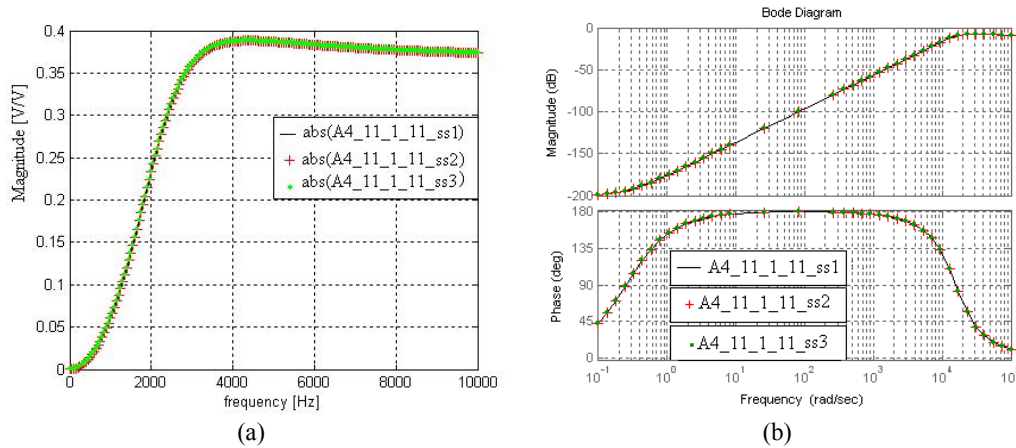


Figure 21: Frequency characteristics of the voltage gain for HPF (a), and Bode characteristics of the voltage gain for HPF (b), by the semi-state approach.

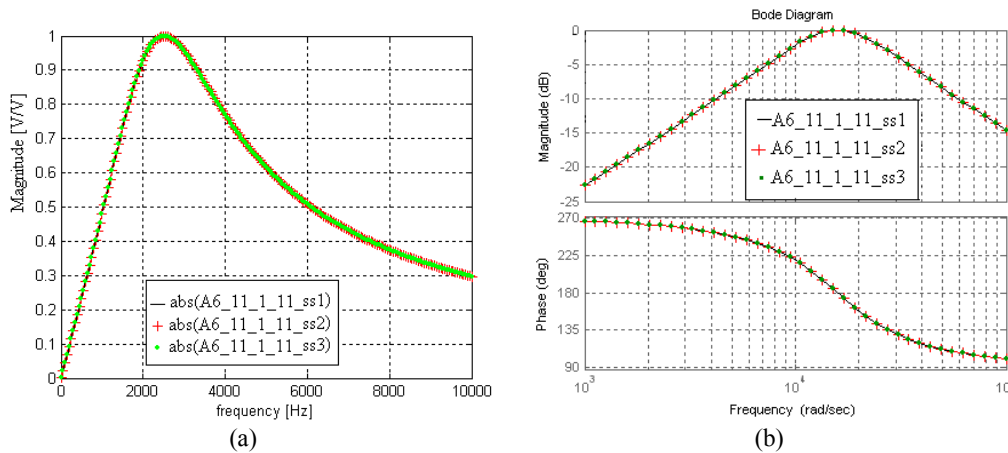


Figure 22: Frequency characteristics of the voltage gain for BPF (a), and Bode characteristics of the voltage gain for BPF (b), by the semi-state approach.

Note: Taking as base the second procedure, we can denote that the computing time for the first procedure is four times bigger, while for the third procedure is twenty times smaller.

Example 9

Consider the tooth filter buffered on 60 Hz represented in **Fig. 23**, in which for the operational amplifiers the linear models are used. In order to verify the above procedures, both in state variable and in semi-state approaches, we consider two input ports (1_8 and 8_2) and one output port – 7_8, and generate the following transfer function matrix:

$$H(s) = \left[A_{7,8,1,8} = \frac{U_{7,8}(s)}{E_{13}(s)} \quad Z_{7,8,8,2} = \frac{U_{7,8}(s)}{J_{14}(s)} \right] \tag{58}$$

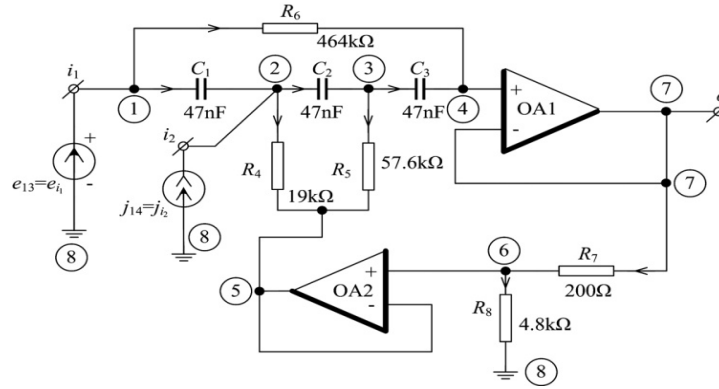


Figure 23: Tooth filter buffered on 60 Hz.

According with the first procedure based on the state equations (26), the transfer function matrix has the following expression:

$$TF_{ST_EQ1} := \left[\frac{1.000 (s + 138.6) (s^2 - 1.139 s + 136800.)}{(s + 132.9) (s^2 + 79.04 s + 142800.)} \quad \frac{0.2128 \cdot 10^8 s^2}{(s + 132.9) (s^2 + 79.04 s + 142800.)} \right]$$

Using the algorithm based on the state equation manipulation in frequency domain, we get:

$$TF_{ST_EQ2} := \left[\frac{0.9996 (s + 138.7) (s^2 - 1.170 s + 136800.)}{(s + 132.9) (s^2 + 78.90 s + 142700.)} \quad \frac{0.04000 (-0.6400 \cdot 10^7 s + 0.1600 \cdot 10^{10} + 0.5320 \cdot 10^9 s^2)}{(s + 132.9) (s^2 + 78.90 s + 142700.)} \right]$$

In the case of the procedures based on the semi-state equations, the circuit function matrix has the following structures:

$$TF_{ss1} := \left[\frac{0.9999 (s + 138.7184) (s^2 - 1.1542 s + 136734.36)}{(s + 132.9879) (s^2 + 78.9378 s + 142626.40)} \quad \frac{0.21276490 \cdot 10^8 s^2}{(s + 132.9879) (s^2 + 78.9378 s + 142626.40)} \right]$$

$$TF_{ss2} := \left[\frac{0.9999 (s + 138.7184) (s^2 - 1.1540 s + 136734.30)}{(s + 132.9879) (s^2 + 78.9379 s + 142626.35)} \quad \frac{0.21276483 \cdot 10^8 s^2}{(s + 132.9879) (s^2 + 78.9379 s + 142626.35)} \right]$$

$$TF_{ss3} := \left[\frac{0.9999 (s + 138.7184) (s^2 - 1.1542 s + 136734.36)}{(s + 132.9879) (s^2 + 78.9378 s + 142626.38)} \quad \frac{0.21276490 \cdot 10^8 s^2}{(s + 132.9879) (s^2 + 78.9378 s + 142626.38)} \right]$$

The frequency characteristics of the voltage-gain and of the transfer impedance, corresponding to these five procedures are plotted in **Figs. 24**, and **25**, respectively.

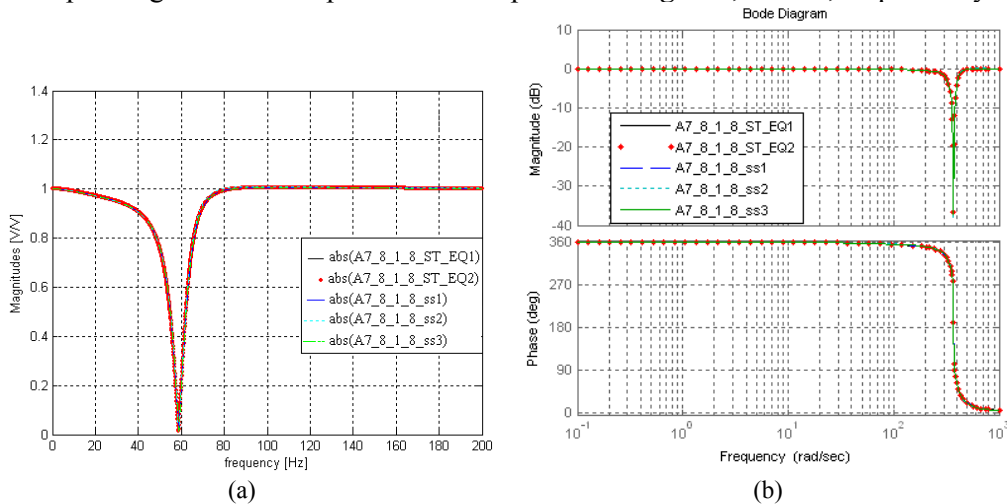


Figure 24: Frequency characteristics of the voltage gain $A_{7,8,1,8}$ (a), and Bode characteristics (b).

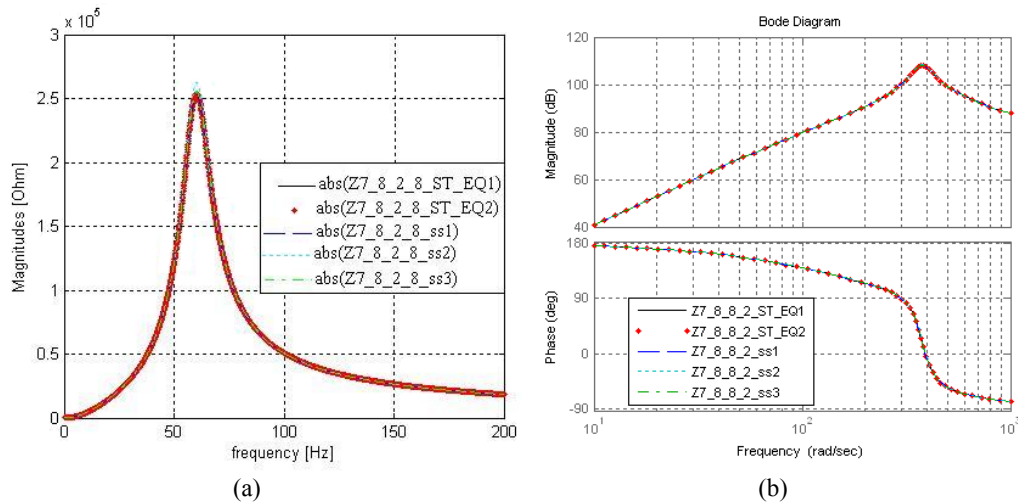


Figure 25: Frequency characteristics of the transfer impedance $Z_{7,8-2}$ (a), and Bode characteristics (b).

We can see that in the above figures the five curves overlap.

The state matrix A of the circuit has the numeric expression (the full symbolic expression is too large to be retained):

$$A := \begin{bmatrix} -105.43733 & 1014.3835 & 1383.7689 \\ -60.633745 & -60.633745 & 308.75161 \\ -45.854785 & -45.854785 & -45.854785 \end{bmatrix}$$

and the eigenvalues are:

$$\text{Eigenvalues} := [-39.468884 + 375.59110 I, -39.468884 - 375.59110 I, -132.98809]$$

Example 10

We want to study the active pass-band filter of the Butterworth type, shown in **Fig. 26(a)**. This filter is designed to extract the 500 Hz harmonic of a periodic signal.

We intend to generate the network functions for two input ports (with e_{29} and j_{30}), and the output ports 21_22 and 13_22. The network function matrix has the form:

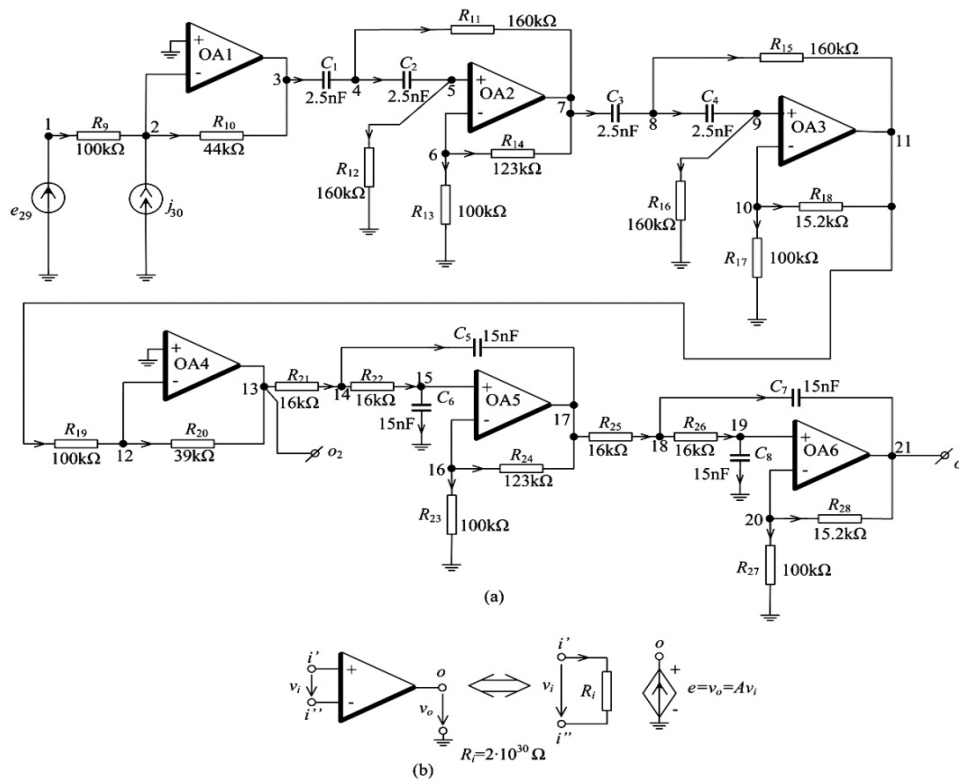


Figure 26: (a) Active band pass filter of the Butterworth type; (b) Linear model for the opamps.

$$H(s) = \begin{bmatrix} A_{21,22_1,22}(s) & Z_{21,22_2,22}(s) \\ A_{13,22_1,22}(s) & Z_{13,22_2,22}(s) \end{bmatrix} \quad (59)$$

The transfer functions were computed by three ways: the two procedures based on the state variables and the algorithm based on the semi-state equation manipulation in frequency domain. Because the computation of the inverse matrices from the equations (54) and (57), having as symbol the complex variable s , is very expensive, these procedures are not used. The frequency characteristics of the four transfer functions corresponding to the three procedures are plotted in Figs. 27 - 30.

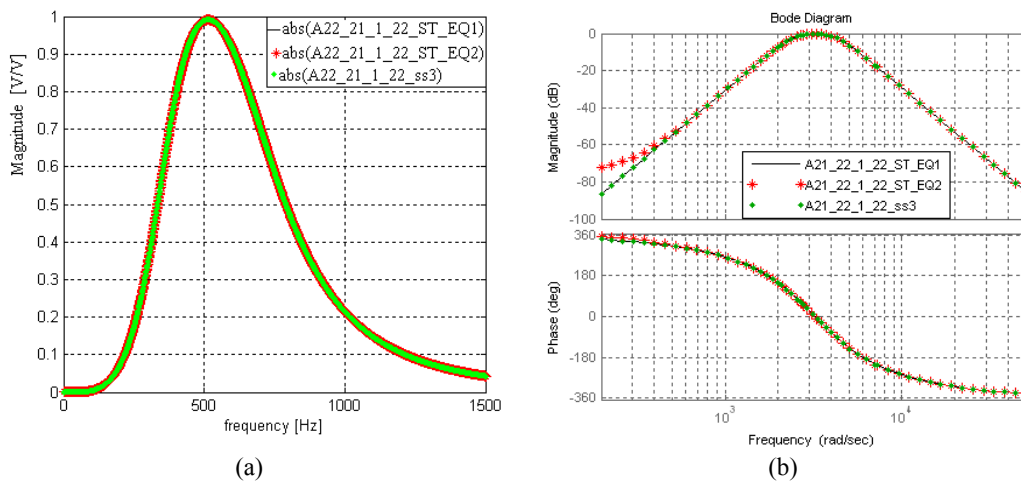


Figure 27: Frequency characteristics of the voltage gain $A_{21,22_1,22}$ (a), and Bode characteristics (b).

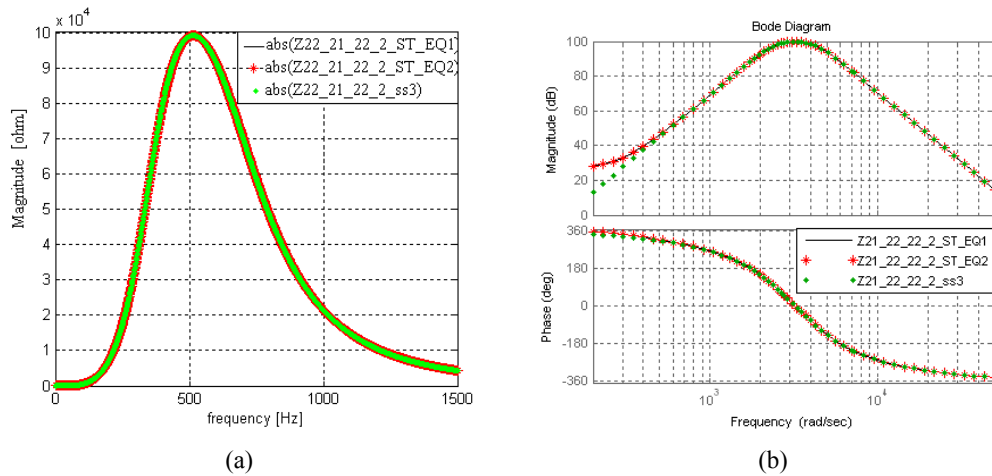


Figure 28: Frequency characteristics of the transfer impedance $Z_{21,22_2,2}$ (a), and Bode characteristics (b).

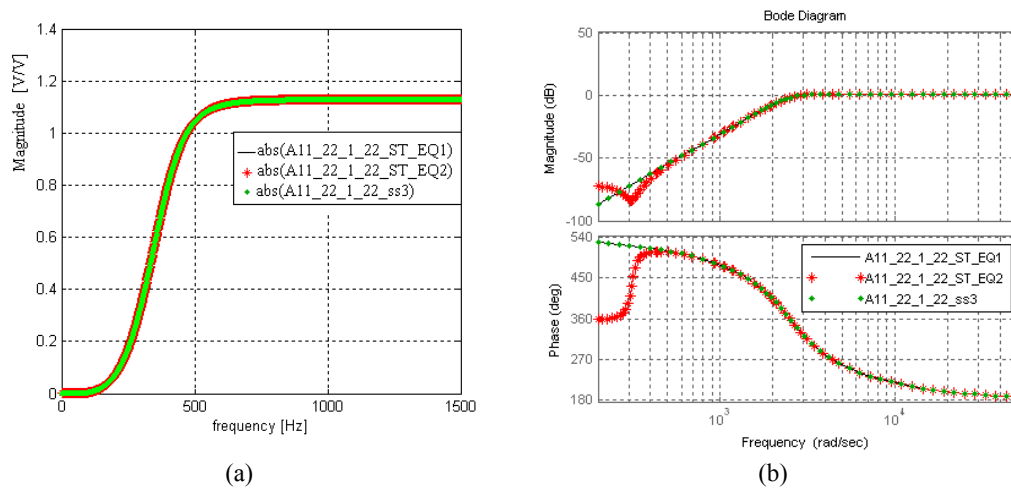


Figure 29: Frequency characteristics of the voltage gain $A_{11,22_1,22}$ (a), and Bode characteristics (b).

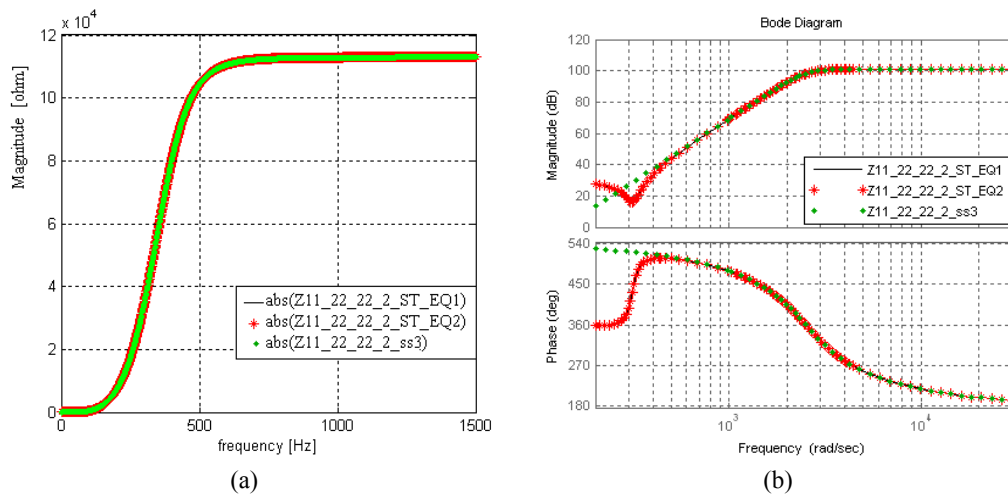


Figure 30: Frequency characteristics of the transfer impedance $Z_{11,22_2,2}$ (a), and Bode characteristics (b).

References

- [1] K. Singhal and J. Vlach, "Symbolic analysis of analog and digital circuits", *IEEE Transactions on Circuits and Systems*, vol. CAS-24, no. 11, pp. 598-609, November 1977.
- [2] S.J. Seda, M. G. R. Degrauwe, and W. Fichtner, "A symbolic analysis tool for analog circuit design automation", *Proc. of IEEE International Conference on Computer-Aided Design*, pp. 488-491, 1988.
- [3] G. Gielen, H. Walscharts, and W. Sansen, "ISAAC: a symbolic simulator for analog integrated circuits", *IEEE Journal of Solid-State Circuits*, vol. 24, no. 6, pp. 1587-1597, December 1989.
- [4] G.M. Wizerba, A. Srivastava, V. Joshi, K.V. Noren, and J.A. Svoboda, "SSPICE -A symbolic SPICE program for linear active circuits", *Proc. of IEEE 32nd Midwest Symposium on Circuits and Systems*, 1989, pp. 1197-1201.
- [5] G. Gielen and W. M. Sansen, *Symbolic analysis for automated design of analog integrated circuits*. New York: Kluwer Academic Publishers, 1991.
- [6] P. M. Lin, *Symbolic network analysis*. New York: Elsevier, 1991.
- [7] M. Amadori, R. Guerrieri, and E. Malavasi, "Symbolic analysis of simplified transfer functions", *Analog Integrated Circuits and Signal Processing*, vol. 3, pp. 9-29, January 1993.
- [8] M.M. Hassoun and K.S. McCarville, "Symbolic analysis of large-scale networks using a hierarchical signal flowgraph approach", *Analog Integrated Circuits and Signal Processing*, vol. 3, pp. 31-42, 1993.
- [9] G. Gielen, P. Wambacq, and W. M. Sansen, "Symbolic analysis methods and applications for analog circuits: a tutorial overview", *Proceedings of the IEEE*, vol. 82, no. 2, pp. 287-303, February 1994.
- [10] J.J. Hsu and C. Sechen, "DC small signal symbolic analysis of large analog integrated circuits", *IEEE Transactions on Circuits and Systems I*, vol. 41, no. 12, pp. 817-828, December 1994.
- [11] R.R. Mielke, "A new signal flowgraph formulation of symbolic network functions", *IEEE Transactions on Circuits and Systems*, vol. CAS-25, pp. 334-340, June 1978.
- [12] L. O. Chua, C. A. Desoer, and E. S. Kuh, *Linear and nonlinear circuits*. New York: McGraw-Hill, 1981.
- [13] F.V. Fernandez, A. Rodriguez-Vasquez, and J.L.Huertas, "Interactive AC modeling and characterization of analog circuits via symbolic analysis", *Analog Integrated Circuits and Signal Processing*, vol. 1, no. 3, pp. 183-208, November 1991.
- [14] W. K. Chen, Editor-in-Chief, *The Circuits and filters handbook*. Boca Raton: CRC Press, 1995.
- [15] J. P. Le Baron and E. Cadran, "Symbolic state equations of linear electronic circuits with degenerates", in *International Workshop on Symbolic Methods and Applications in Circuit Design*, 1998, pp. 80-86.
- [16] P. Wambacq, F.V. Fernandez, G. Gielen, W. Sansen, and A. Rodriguez-Vazquez, "Algorithm for efficient symbolic analysis of large analogue circuits", *Electronics Letters*, pp.1108-1109, July 1994.
- [17] P. Wambacq, F.V. Fernandez, G. Gielen, W. Sansen, and A. Rodriguez-Vazquez, "Efficient symbolic computation of approximated small-signal characteristics of analog integrated circuits", *IEEE Journal of Solid-State Circuits*, vol. 30, no. 3, pp. 327-330, March 1995.
- [18] Q. Yu and C. Sechen, "A unified approach to the approximate symbolic analysis of large analog integrated circuits", *IEEE Transactions on Circuits and Systems – I*, vol. CAS-43, no. 8, pp. 656-669, August 1996.
- [19] L. Dumitriu, M. Iordache, R. Muntean, and R. Botinaț, "Efficient generation of symbolic network functions using two-graph decomposition on levels", in *International Workshop on Symbolic Methods and Applications in Circuit Design*, 1998, pp. 191-198.
- [20] M. Iordache, Lucia Dumitriu, R. Muntean, and R. Botinaț, "An algorithm for finding all spanning trees in increasing weight order", in *International Workshop on Symbolic Methods and Applications in Circuit Design*, 1998, pp. 99-105.
- [21] O. Guerra, E. Roca, F.V. Fernandez, and A. Rodriguez-Vazquez, "Approximate symbolic analysis of hierarchically decomposed analog circuits", *Analog Integrated Circuits and Signal Processing*, vol. 31, pp. 131-145, May 2002.
- [22] O. Guerra, J.D. Rodriguez-Garcia, F.V. Fernandez, and A. Rodriguez-Vazquez, "A symbolic pole/zero extraction methodology based on analysis of circuit time-constants", *Analog Integrated Circuits and Signal Processing*, vol. 31, pp. 101-118, May 2002.
- [23] M. Pierzchala and B. Rodansky, "Two-graph stamps for linear controlled sources", in *International Workshop on Symbolic Methods and Applications in Circuit Design*, 2004, pp. 44-47.

- [24] M. Iordache and Lucia Dumitriu, "The generalized topological formula for transfer function generation by two-graph tree enumeration", *Analog Integrated Circuits and Signal Processing*, vol. 47, no. 1, pp.85-100, April 2006.
- [25] G.E. Alderson and P.M. Lin, "Computer generation of symbolic network functions -a new theory and implementation", *IEEE Transactions on Circuit Theory*, vol. CT-20, pp. 48-56, January 1973.
- [26] P. Sannuti and N.N. Puri, "Symbolic network analysis -An algebraic formulation", *IEEE Transactions on Circuits and Systems*, vol. CAS-27, pp. 679-687, August 1980.
- [27] K. Singhal and J. Vlach, "Generation of immittance functions in symbolic form for lumped distributed active networks", *IEEE Transactions on Circuits and Systems*, vol. CAS-21, no. 1, pp. 57-67, January 1974.
- [28] I. Garcia-Vargas, M. Galan, F.V. Fernandez, and A. Rodriguez-Vazquez, "New algorithms for reference generation in symbolic analysis of large analog circuits", *Proc. of European Design and Test Conference*, pp. 395-399, 1997.
- [29] F.V. Fernandez, O. Guerra, J.D. Rodriguez-Garcia, and A. Rodriguez-Vazquez, "Symbolic analysis of analog integrated circuits: the numerical reference generation problem", *IEEE Transactions on Circuits and Systems -II*, vol. 45, no. 10, pp.1351-1361, October 1998.
- [30] L. O. Chua, and P. M. Lin, *Computer-aided analysis of electronic circuits*. Englewood Cliffs, N. J.: Prentice Hall, Inc., 1975.
- [31] A. E. Schwarz, *Computer-aided design of microelectronic circuits and systems*. London: Academic Press, 1987.
- [32] J. Vlach and K. Singhal, *Computer methods for circuit analysis and design*. 2nd ed., New York: Van Nostrand Reinhold, 1994.
- [33] M. Iordache, and Lucia Dumitriu, "Symbolic state equations for analog circuits", *Revue Roumaine des Sciences Techniques, Série Électrotechnique et Énergétique*, Vol. 45, no. 1, pp. 21-31, January 2000.
- [34] M. Iordache and Lucia Dumitriu, "Computer formulation of symbolic state equations for analog nonlinear circuits with excess elements", in *International Symposium on Nonlinear Theory and its Applications*, 2000, pp. 355-358.
- [35] M. Iordache and Lucia Dumitriu, "A decomposition technique for setting up the symbolic state equations of large-scale analog circuits", *Proc. of European Conference on Circuit Theory and Design*, 2001, pp. 201-204.
- [36] M. Iordache and Lucia Dumitriu, "Efficient decomposition techniques for symbolic analysis of large-scale analog circuits by state variable method", *Analog Integrated Circuits and Signal Processing*, vol. 40, no. 3, pp.235-253, September 2004.
- [37] Angela M. Hodge and R. W. Newcomb, "Semistate theory and analog VLSI design", *IEEE Circuit and Systems Magazine*, vol. 2, no. 2, pp.30-49, Second Quarter 2002.
- [38] M. Iordache, Lucia Dumitriu, and L. Mandache, "Time-domain modified nodal analysis for large-scale analog circuits", *Revue Roumaine des Sciences Techniques, Série Électrotechnique et Énergétique*, Tome 48, No. 2-3, pp. 257-268, 2003.
- [39] M. Iordache, and Lucia Dumitriu, "Network function generation for MIMO systems", in *International Conference on Optimization of Electrical and Electronic Equipment*, 2008, pp. 57-62.
- [40] E. Hennig, *Symbolic approximation and modeling techniques for analysis and design of analog circuit*. Aachen: Shaker Verlag, 2000.
- [41] M. Iordache, Lucia Dumitriu, "Time domain diakoptic analysis based on reduced-order state equations", *International Journal of Bifurcation and Chaos*, Volume: 17, Issue: 10, pp. 3625-3631, October 2007.
- [42] E. Hennig, M. Wiese, and R. Sommer, "Symbolic pole/zero approximation using eigenvalue shift prediction", in *Workshop on Symbolic Methods and Applications to Circuit Design*, 1998, pp. 29-35.
- [43] Q. Yu and C. Sechen, "Approximate symbolic analysis of large analog integrated circuits", in *European Conference on Circuit Theory and Design*, 1995, pp. 67-70.
- [44] F. V. Fernandez, A. Rodriguez-Vazquez, and J. L. Huertas, "A tool for symbolic analysis of analog integrated circuits including pole/zero extraction", in *European Conference on Circuit Theory and Design*, 1991, pp. 752-761.
- [45] J. Hsu and C. Sechen, "The shifting approach to symbolic analysis of large analog integrated circuits", in *International Workshop on Symbolic Methods and Applications in Circuit Design*, 1994, pp.211-230.

CHAPTER 5**Symbolic Analysis of Analog Circuits by Flow-Graphs****Mourad Fakhfakh^{1,*}, Irina Asenova² and Mourad Loulou¹**¹University of Sfax, Tunisia and ²Higher school of transport, Sofia, Bulgaria

Abstract: This chapter details the use of *Coates* flow-graphs for the analysis of analog circuits. Basic concepts of the association of graphs to algebraic equations are presented. Solving the flow-graph by topological methods is detailed. Application to the computation of symbolic transfer functions of analog circuits is highlighted. In addition, application to the generation of flow-graph stamps for some circuits, such as, controlled sources, MOS transistors, current conveyors, is also presented, and some application examples for computing transfer function of current conveyor based filters are proposed. Further, first-order symbolic sensitivity analysis of nullor based networks is described. Generation of partial symbolic transfer functions, using modified *Coates* flow-graphs, is detailed. Examples illustrating the proposed method are presented.

Keywords: Coates flow-graphs, modified coates flow-graphs, symbolic analysis, nullors, flow-graph stamps, symbolic sensitivity analysis, controlled sources, MOS transistors, current conveyors, DVCCII, FDCCII.

1. Introduction

Flow-graphs can be used as tools for the analysis, modeling and synthesis in network theory and in linear system theory.

One important step completing every network analysis and synthesis procedure is the determination of the transfer functions and the sensitivity of the realized structure. It is well known that the analysis (including sensitivity determination) of the synthesized circuit can be implemented on the basis of a suitable flow-graph representation of the variable relationships.

The modified *Coates* flow-graphs allow in an easy way the analysis of active networks by using nullor models of the active devices. Main advantages of symbolic analysis [1], and in particular the symbolic sensitivity analysis, are summarized as follows:

- Repeated analysis is not needed and particular derivatives can be avoided,
- Determination of transfer functions and first-order sensitivity for different frequencies can be performed,
- Round-off errors are avoided,
- Particular effects of the (circuit's) parameters can be studied.

The chapter comprises three main sections. Section 2 discusses the basic concepts of *Coates* flow-graphs. Their association with algebraic equations is given. This section

*Address correspondence to Mourad Fakhfakh: University of Sfax, Tunisia; E-mail: mourad.fakhfakh@ieee.org

describes the flow-graph based method for generating the symbolic transfer functions of analog circuits. Section 3 gives application to the generation of flow-graph stamps for circuits, such as, controlled sources, MOS transistors, current conveyors. Further, some application examples for computing transfer functions of current conveyor based filters are proposed. Section 4 discusses determination of first-order symbolic sensitivity by using the nullor model and the modified *Coates* flow-graph. In order to reflect the nullator-norator pair influence on the network transfer functions, rules for transformations of flow-graphs are detailed. Generation of partial symbolic transfer functions, obtained using modified *Coates* flow-graphs, is presented. Op-Amp based applications are given.

2. Basic Concepts of *Coates* Flow-Graphs

The purpose of associating a graph to a set of linear algebraic equations is to solve it by a topological method. The first idea belongs to *Mason* [2]; He introduced the signal-flow-graphs. Then, in [3] *Coates* proposed representing the set of equations by a flow-graph that depends only on the algebraic structure of the set of equations.

Expression (1) presents the general form of a matrix system of a set of linear equations.

$$A X - B = O \quad (1)$$

$A = (a_{ij})$, $X = (x_j)$ and $B = (b_i)$ represent the adjacency matrix, the variables' vector and the input vector, respectively.

The algebraic solution of (1) is generally obtained using the following expression:

$$x_j = \frac{\sum_i b_i \Delta_{ij}}{\Delta} \quad (2)$$

The link between a determinant of a matrix and the associated graph is obtained by referring to expression (3).

$$\Delta = \sum_{\{j\}} \varepsilon_{j_1 j_2 \dots j_n} a_{1j_1} a_{2j_2} \dots a_{nj_n} \quad (3)$$

Representing the aforementioned expression by a flow-graph (G) consists of associating vertices to the variables and linking these vertices by weighted edges. The weights correspond to the adjacency variables. Supplementary vertices are added to the graph to represent the sources.

Fig. 1 illustrates the case of a matrix system (expression (4)) with three variables.

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} \quad (4)$$

Coates proved that expression (2) can be evaluated in a topological way. For this purpose, it is useful to introduce the following definitions:

- A one Factor (1F) is a *Hamiltonian* [4] directed sub-graph of (G). *i.e.* a traceable cycle that visits all the graph vertices exactly once and returns to the starting vertex,
- A one Factorial connection (1FC_{ij}) from a vertex *i* to a vertex *j* is a spanning sub-graph of (G) that contains a directed path from *i* to *j* and a set of disjoint vertices of traceable directed cycles comprising all graph vertices, but those included in the *i-j* directed path.

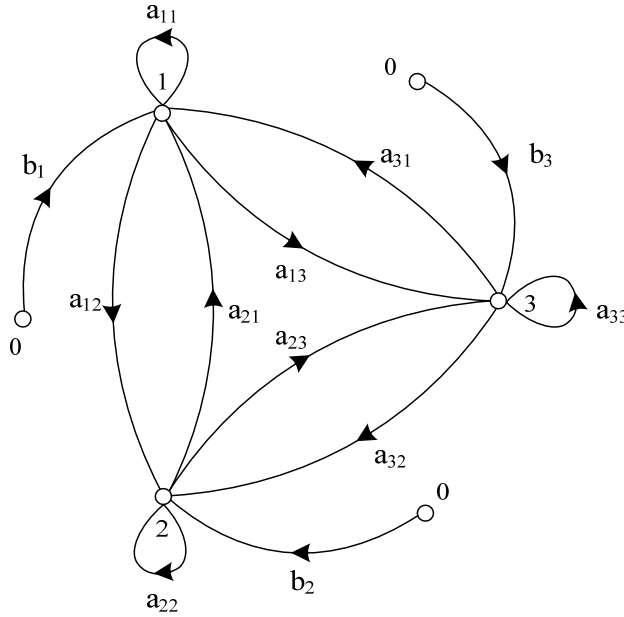


Figure 1: Flow-graph associated to the matrix system given in (4).

Expressions (5) and (6) give the topological evaluation of the determinants of expression (2) [5]:

$$\Delta = \sum_{1F} (-1)^{EC_{1F}-1} f(1F) \tag{5}$$

$$\Delta_{ij} = \sum_{1FC_{ij}} (-1)^{EC_{1FC_{ij}}-1} f(1FC_{ij}) \tag{6}$$

EC_{1F} and EC_{1FC} represent the number of components, which have an even degree, in the 1F and 1FC, respectively. *i.e.* connected sub-graphs containing an even number of edges (notice that an isolated node is an even component). Function *f* performs the product of all transmittances of the 1F (1FC_{ij}).

Cofactors Δ_{ij} represent the minor obtained by eliminating the *i*th row and the *j*th column from matrix *A*. The number of 1F is equal to the permanent of the matrix associated to *A*, whereas the number of 1FC_{ij} equals the permanent of the minor obtained by eliminating the *i*th row and the *j*th column from *A*.

In order to illustrate the proposed approach, let's consider the matrix system (4).

1Fs and 1FCs of the corresponding graph are presented in **Figs. 2** and **3**.

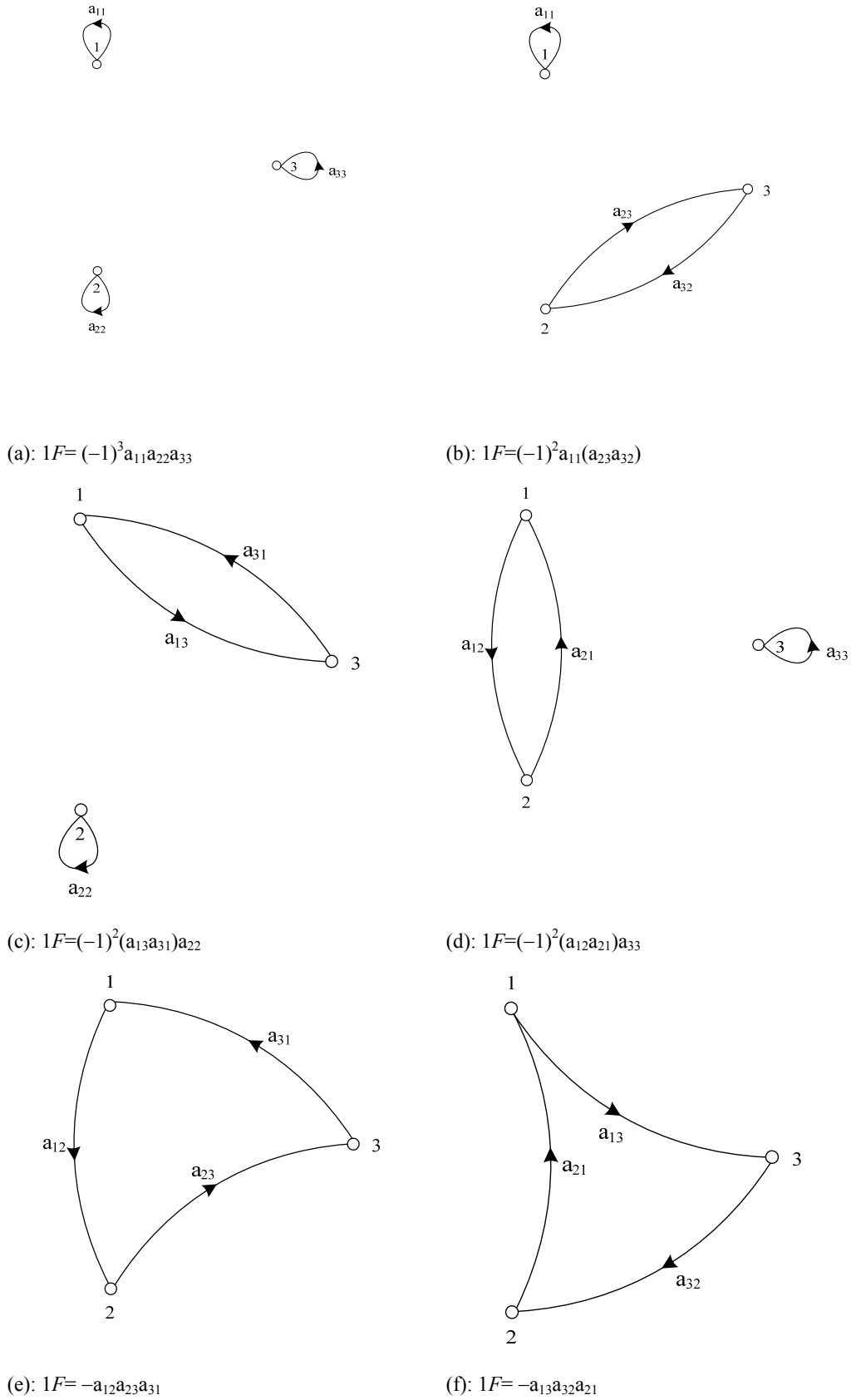
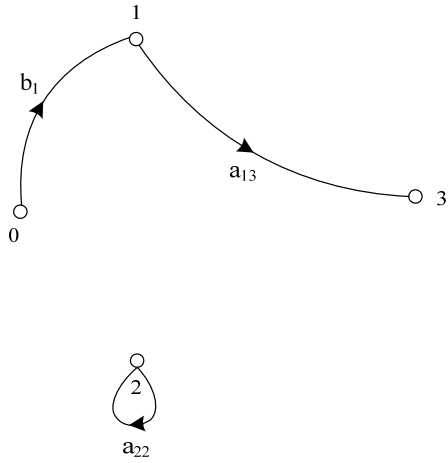
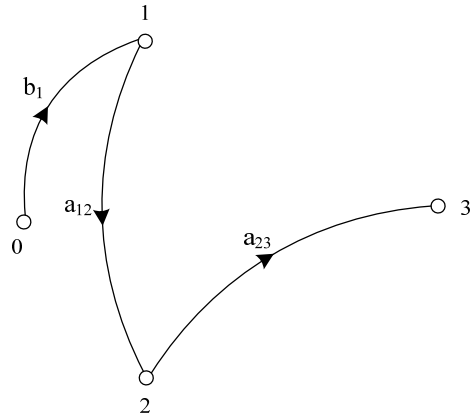


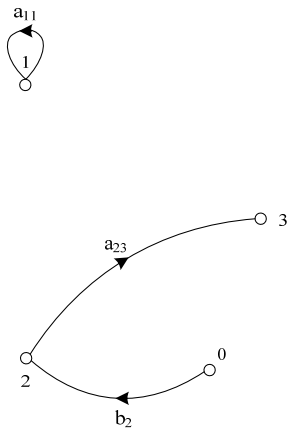
Figure 2: 1Fs of the graph presented in Fig. 1.



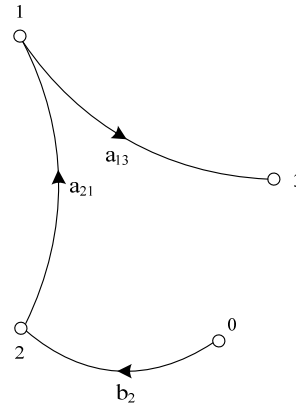
(a): $1FC_{03} = (-1)b_1 a_{13} a_{22}$



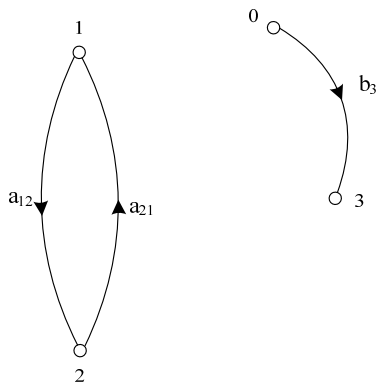
(b): $1FC_{03} = b_1 a_{12} a_{23}$



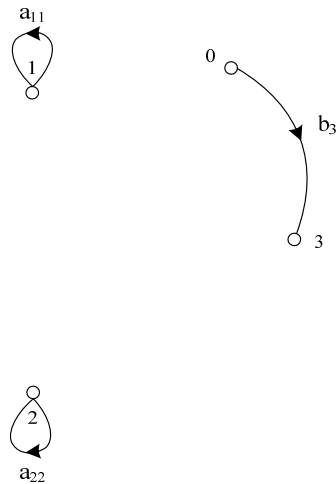
(c): $1FC_{03} = (-1)b_2 a_{23} a_{11}$



(d): $1FC_{03} = b_2 a_{21} a_{13}$



(e): $1FC_{03} = (-1)b_3 a_{21} a_{12}$



(f): $1FC_{03} = b_3 a_{22} a_{11}$

Figure 3: $1FC_{03}$ of the graph presented in Fig. 1 (for $j=3$).

In the following we present some application examples for computing symbolic transfer functions of analog circuits.

3. Application Examples

3.1. A passive filter

Fig. 4 presents a passive filter whose flow-graph can be represented by the graph given in **Fig. 5**. The vertices are associated to the circuit's nodes. **Figs. 6** and **7** give expressions of $1F$ and $1FC_{14}$ of the flow-graph of **Fig. 5**.

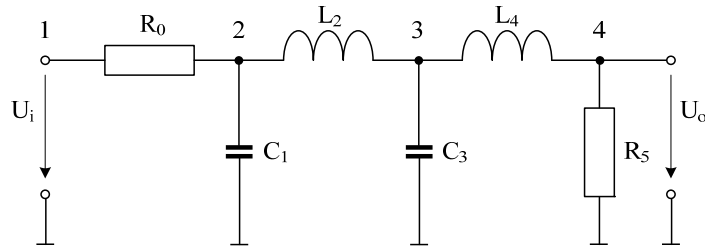


Figure 4: A passive (Cauer) filter.

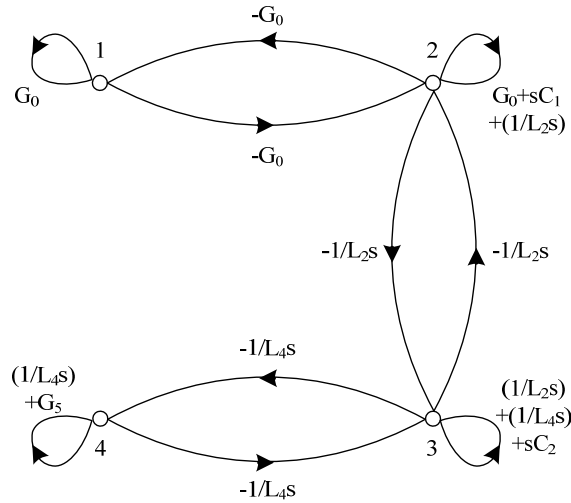


Figure 5: The flow-graph associated to the circuit of **Fig. 4**.

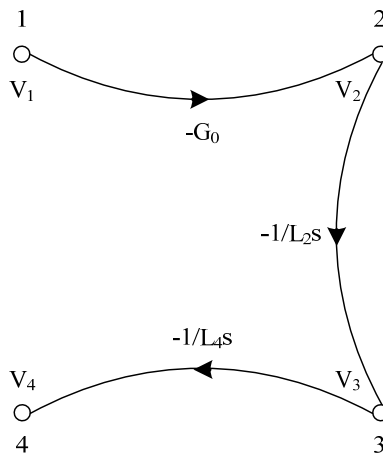


Figure 6: The unique $1F$ of the graph of **Fig. 5**: $1FC_{14} = -G_0(1/sL_2)(1/sL_4)$.

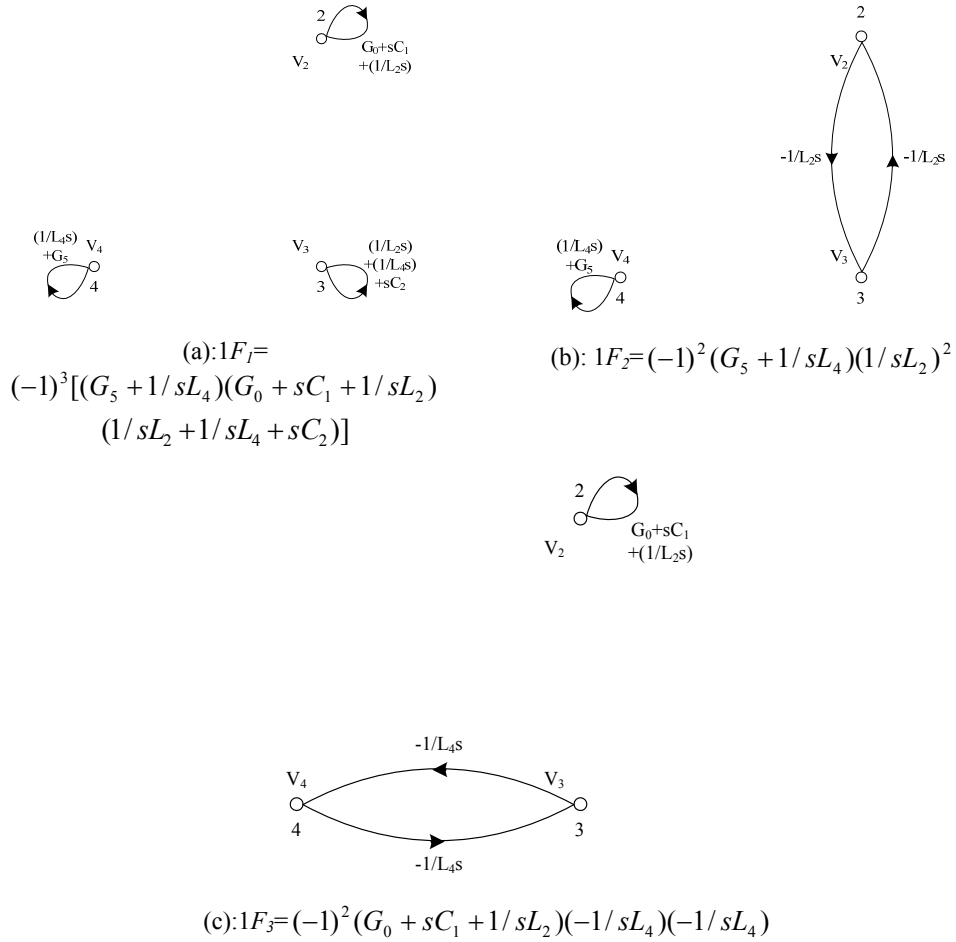


Figure 7: The 1FCs of the graph of Fig. 5.

Accordingly, we have:

$$\frac{U_o}{U_i} = \frac{1FC_{14}}{\sum_{i=1}^3 1F_i} \tag{7}$$

3.2. MOS circuits

Flow-graph models for controlled circuits can be found in [6, 7]. For instance, Fig. 8 presents the graph corresponding to a VCCS.

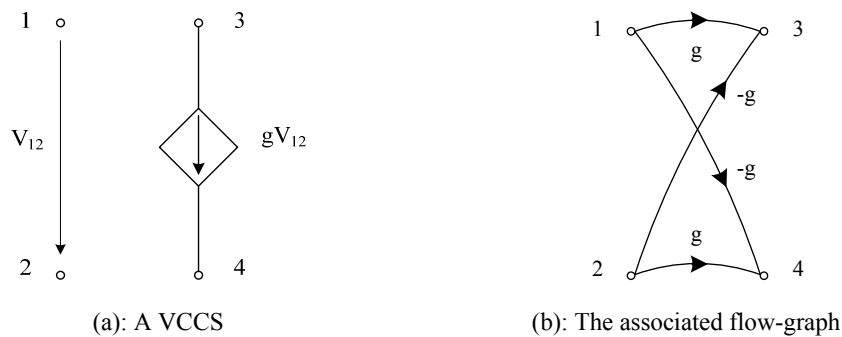


Figure 8: A voltage controlled current source.

Accordingly, a flow-graph model of a MOS transistor can be established, as it is presented in **Fig. 9**.

C_{gs} , g_{ds} and g_m are the gate to source capacitance, the conductance and the transconductance of a MOS transistor, respectively. Vertices 1, 2 and 3 correspond to the gate, the source and the drain of the MOS transistor, respectively.

It is to be noticed that the complexity of the model can be adjusted by adding/removing parasitic components, such as the drain to source capacitance, the bulk vertex, etc.

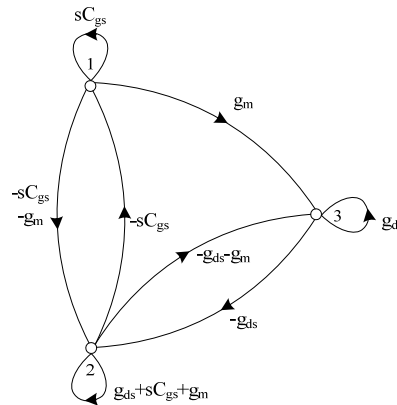


Figure 9: A flow-graph of a MOS transistor ((g_m, g_{ds}, C_{gs}) model).

Example 1 **Fig. 10** presents a MOS amplifier. The equivalent flow-graph is presented in **Fig. 11**. It is to be noticed that the (C_{gs}, g_{ds}, g_m) equivalent model is used for the MOS transistors.

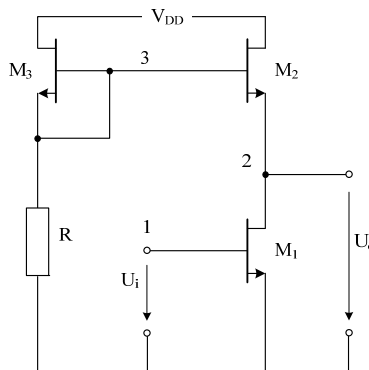


Figure 10: A MOS amplifier.

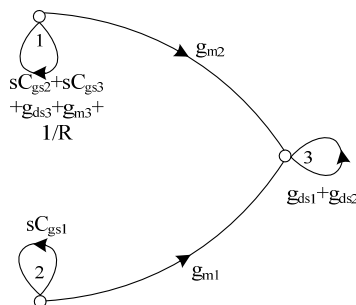


Figure 11: The flow-graph of the circuit presented in **Fig. 10** (considering the (g_m, g_{ds}, C_{gs}) model).

It is easy to compute the $1F$ and the $1FC_{12}$ of the flow-graph of **Fig. 11**, they are given by expressions (8) and (9), respectively.

$$1F = (g_{ds3} + g_{m3} + sC_{gs2} + sC_{gs3} + 1/R)(g_{ds1} + g_{ds2}) \tag{8}$$

$$1FC_{12} = -g_{m1}(sC_{gs2} + sC_{gs3} + g_{ds3} + g_{m3} + 1/R) \tag{9}$$

Thus, the transfer function is:

$$\frac{U_o}{U_i} = - \frac{g_{m1}}{g_{ds1} + g_{ds2}} \tag{10}$$

Example 2 Flow-graphs can be used for didactic purposes. The following example (**Fig. 12**) presents a MOS regulated cascode current mirror. It is easy, using flow-graphs to compute parameters (conductance and transconductance) of the equivalent MOS transistor. **Fig. 13** presents the corresponding flow-graph. In order to compute the equivalent transconductance, node 4 is grounded and we have $g_{meq} = I_{CC}/U_i \cdot I_{CC}$ is the shortcut current flowing through node 4. Expressions (11) and (12) present the $1F$ and $1FC_{13}$ associated to the simplified graph (without node 4).

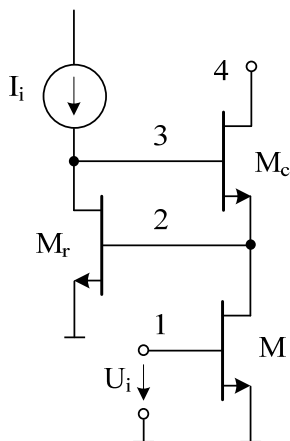


Figure 12: Regulated cascode current mirror.

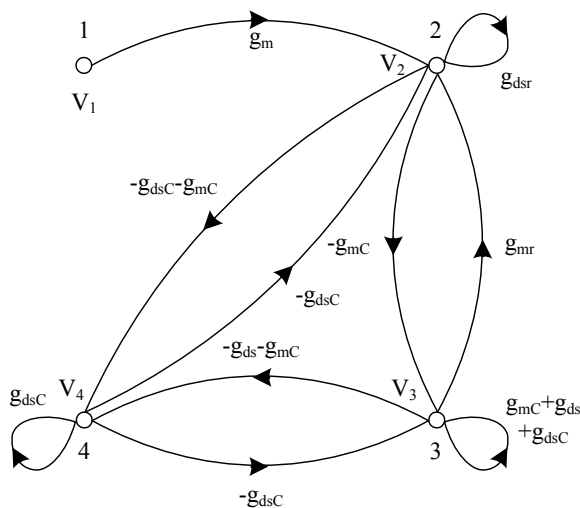


Figure 13: The flow-graph associated to the circuit of **Fig. 12**.

$$\sum 1F = g_{mr}g_{mC} + g_{dsr}(g_{mC} + g_{dsC} + g_{ds}) \quad (11)$$

$$1FC_{13} = \frac{-g_m g_{dsr}}{g_{mr}g_{mC} + (g_{mC} + g_{dsC} + g_{ds})g_{dsr}} \quad (12)$$

Knowing that I_{CC} is the drain current of transistor M, we have:

$$I_{CC} = g_m V_1 + g_{ds} V_3 \quad (13)$$

Finally, the equivalent transconductance is obtained, as given by expression (14).

$$\frac{I_{CC}}{V_1} = \frac{g_m g_{mr} g_{mC} + (g_{mC} + g_{dsC})g_{dsr} g_m}{g_{mr} g_{mC} + (g_{mC} + g_{dsC} + g_{ds})g_{dsr}} \approx g_m \quad (14)$$

In order to compute the equivalent conductance, node 1 is grounded and node 4 becomes the input one.

$$1F = g_{dsr} g_{ds} \quad (15)$$

$$1FC_{34} = g_{mC} g_{mr} + g_{dsr}(g_{ds} + g_{mC} + g_{dsC}) \quad (16)$$

$$\frac{V_3}{V_4} = \frac{g_{dsr} g_{dsC}}{g_{mC} g_{mr} + g_{dsr}(g_{ds} + g_{mC} + g_{dsC})} \approx \frac{g_{dsr} g_{dsC}}{g_{mC} g_{mr}} \quad (17)$$

Expressions (15) and (16) present the corresponding $1F$ and $1FC_{34}$. Finally, the equivalent transconductance is given by expression (17).

3.3. Current conveyor based circuits

Biolek and Biolkova proposed in [6] a matrix stamp and *Coates* graph stamp of *Thevenin* model of reciprocal two port circuits. On the base of this model, a graph stamp for current conveyors was proposed [6]. **Fig. 14** shows the *Thevenin* model of the two-port circuit and its flow-graph stamp.



Figure 14: (a) A two-port graph model, (b) Its graph stamp.

Example 1 On the base of the basic circuit of **Fig. 14(a)** and its corresponding graph, a stamp for three-port Current Conveyors (CC) was established (**Fig. 14(b)**) [6]. **Figs. 15(a, b)** present the CC block diagram and its corresponding graph stamp, respectively. Its terminal relations are given by expression (18), where α , β and γ define the type of the CC [8]. *i.e.* if $\alpha=1$, the CC is a first generation one. If $\alpha=0$, it is a second generation one. $\beta=-1$ or $+1$ whether the CC is an inverting or a non-inverting one [9]. Finally, γ indicates if the CC is a positive ($\gamma=+1$) or a negative ($\gamma=-1$) one.

$$\begin{pmatrix} i_y \\ v_x \\ i_z \end{pmatrix} = \begin{pmatrix} 0 & \alpha & 0 \\ \beta & 0 & 0 \\ 0 & \gamma & 0 \end{pmatrix} \begin{pmatrix} v_y \\ i_x \\ v_x \end{pmatrix} \tag{18}$$

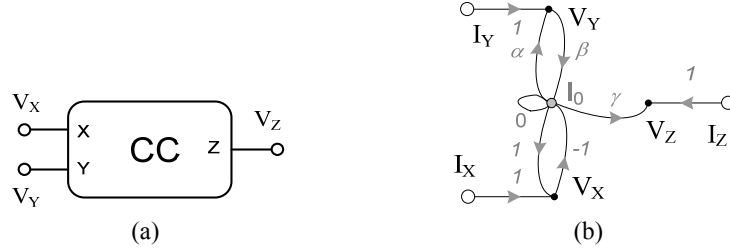


Figure 15: (a) Block diagram of a CC, (b) its flow-graph.

The circuit presented in Fig. 16(a) is a voltage-mode notch, low-pass and band-pass filter [10]. It is designed using four positive non-inverting second generation current conveyors (CCII+). In order to determine the transfer functions, the flow-graph of the filter was constructed. It is presented in Fig. 16(b).

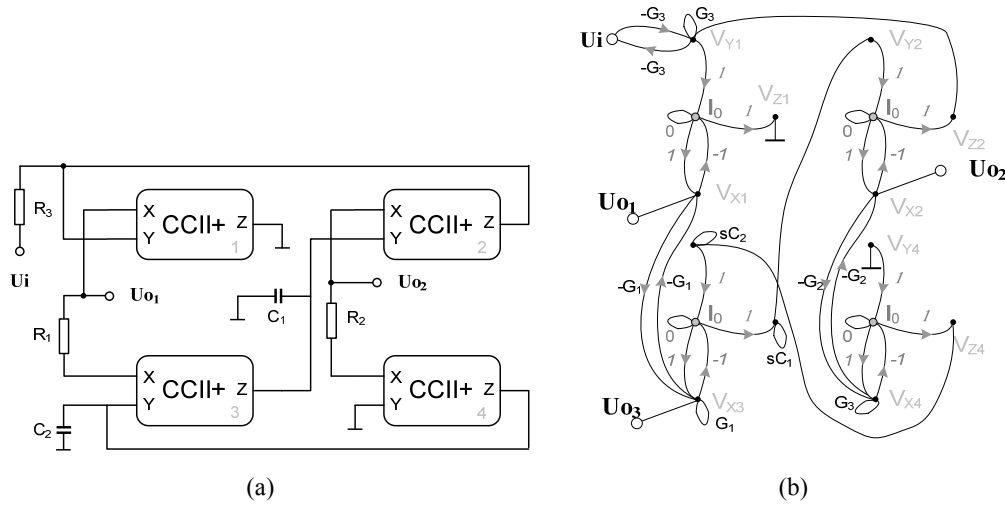


Figure 16: A multifunction voltage mode CCII+ based filter and the corresponding flow-graph.

$$\frac{U_{o1}}{U_i} = \frac{s^2 C_1 C_2 G_3 + G_1 G_2 G_3}{s^2 C_1 C_2 G_3 + s C_2 G_1 G_2 + G_1 G_2 G_3} \tag{19}$$

$$\frac{U_{o2}}{U_i} = \frac{-s C_2 G_3 G_1}{s^2 C_1 C_2 G_3 + s C_2 G_1 G_2 + G_1 G_2 G_3} \tag{20}$$

$$\frac{U_{o3}}{U_i} = \frac{G_1 G_2 G_3}{s^2 C_1 C_2 G_3 + s C_2 G_1 G_2 + G_1 G_2 G_3} \tag{21}$$

Expressions (19), (20) and (21) present transfer functions U_{o1}/U_i , U_{o2}/U_i and U_{o3}/U_i , respectively. Obtained expressions are conforming to those presented in [10].

Example 2 A DVCCII is a five-port building block [11,12]. The DVCCII is defined by expression (22). Its block diagram and its flow-graph are presented in Figs. 17(a, b), respectively [13].

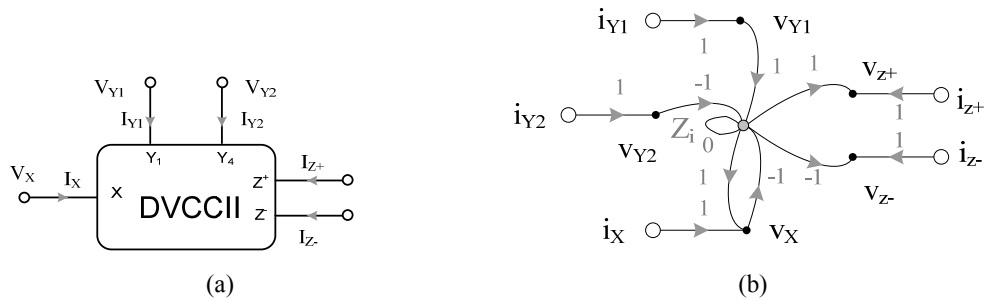


Figure 17: (a) Block diagram of a DVCCII, (b) its flow-graph stamp.

$$\begin{pmatrix} i_{Y1} \\ i_{Y2} \\ v_X \\ i_{Z+} \\ i_{Z-} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{pmatrix} \begin{pmatrix} v_{Y1} \\ v_{Y2} \\ i_X \\ v_{Z+} \\ v_{Z-} \end{pmatrix} \quad (22)$$

A Multiple-Input Single-Output BP-LP second order filter designed using two DVCCII is presented in Fig. 18(a) [12]. The flow-graph corresponding to this filter is shown in Fig. 18(b). Applying the Coates' technique allows computing both transfer functions: \$U_o/U_{i_1}\$ and \$U_o/U_{i_2}\$. Their expressions are given by (23) and (24), respectively. They are conforming to the expression given in [12].

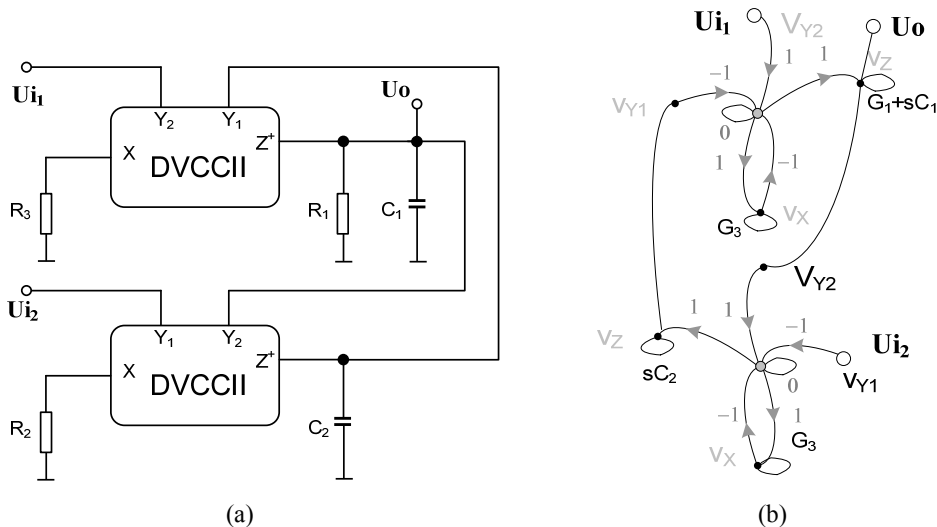


Figure 18: (a) A multifunction DVCCII based filter, (b) its flow-graph.

$$\frac{U_o}{U_{i_1}} = \frac{-sGC_2}{s^2C_1C_2 + sG_1C_2 + G_1G_2} \quad (23)$$

$$\frac{U_o}{U_{i_2}} = \frac{-G_1G_2}{s^2C_1C_2 + sG_1C_2 + G_1G_2} \quad (24)$$

Example 3 A FDCCII is an eight-port building block [14], which is defined by expression (25). Fig. 19(a) represents the FDCCII block diagram. The proposed flow-graph stamp of the FDCCII is presented by Fig. 19(b) [13].

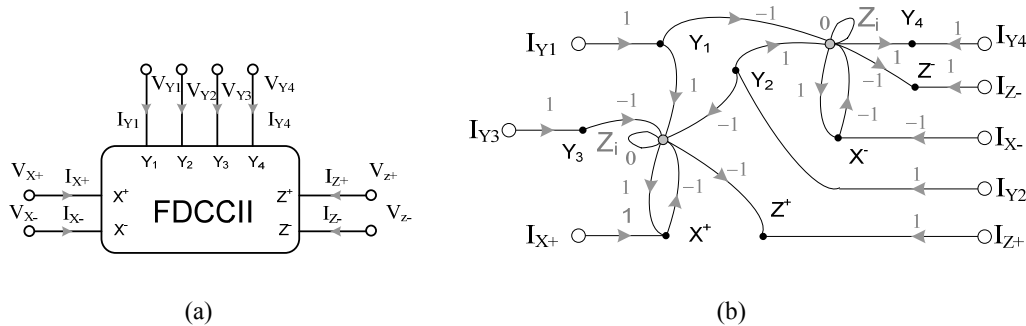


Figure 19: (a) Block diagram of the FDCCII, (b) its flow-graph.

$$\begin{pmatrix} v_{X+} \\ v_{X-} \\ i_{Z+} \\ i_{Z-} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} i_{X+} \\ i_{X-} \\ v_{Y1} \\ v_{Y2} \\ v_{Y3} \\ v_{Y4} \end{pmatrix} \quad (25)$$

Fig. 20(a) presents a high-input and low-output impedance universal voltage mode filter designed using two FDCCII [12]. The corresponding flow-graph is presented by Fig. 20(b). The graph resolution gives the transfer functions given by expressions (26), (27), and (28). These expressions are conforming to those presented in [12].

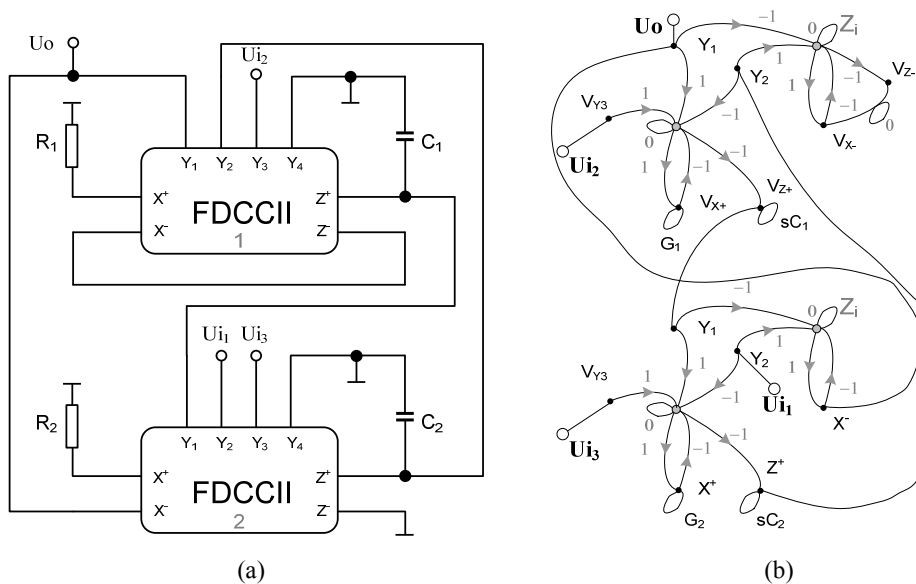


Figure 20: An FDCCII based universal voltage mode filter and its flow-graph.

$$\frac{U_0}{U_1} = \frac{s^2 C_1 C_2}{s^2 C_1 C_2 + s C_2 G_1 + G_1 G_2} \quad (26)$$

$$\frac{U_0}{U_2} = \frac{-s C_2 G_1}{s^2 C_1 C_2 + s C_2 G_1 + G_1 G_2} \quad (27)$$

$$\frac{U_0}{U_3} = \frac{G_1 G_2}{s^2 C_1 C_2 + s C_2 G_1 + G_1 G_2} \quad (28)$$

Finally, it is to be highlighted that the *Coates* flow-graph approach for computing symbolic transfer functions of analog circuits was implemented in MATLAB and CASCADES.1 software was proposed in [15]. Thanks to CASCADES.1, the user has only to draw his circuit's schema and specifies the input and output nodes. The MATLAB program calls the netlist and automatically computes and returns the numerator, the denominator and the full symbolic transfer function. Further, CASCADES.1 allows modifying the complexity of the adopted equivalent models (for MOS transistors).

4. Determination of First-Order Symbolic Sensitivity by Using the Nullor Model and the Modified *Coates* Flow-Graph

One important step completing every network analysis and synthesis procedure is the determination of the sensitivity of the considered structure. Haigh *et al.* [16] proved a theorem that determinates the basis, consisting of passive elements and only one of which is a nullor. The latter is used as a universal active element for an equivalent representation of an active network [17-21]. On the other hand the analysis of a synthesized network can be implemented and simplified on the base of the modified *Coates* flow-graph [22], using some network partial transfer functions [23].

References [23, 24] describe a symbolic method of determination of the first-order transfer function sensitivity in electrical networks. Their equivalent nullor circuits present a starting point for the sensitivity analysis. It is based on some transformations of the modified *Coates* flow-graph in order to reflect the nullator-nullorator pair influence in the network.

This method of sensitivity determination of linear networks is applied. Compared with other methods [25], its main advantages are:

- A simple way of obtaining of an equivalent nullor circuit, representing all active elements;
- Repeated analysis are not necessary to be performed.

The sequence of presentation of the proposed method is:

1. Generation of an equivalent nullor representation of the active network,
2. Modification of the flow-graph by taking into account effects of nullors,
3. Determination of the first-order symbolic transfer function sensitivity with respect to a transmission coefficient in the flow-graph.

4.1. Determination of first-order symbolic transfer function sensitivity with respect to a passive element

4.1.1. An equivalent nullor network

According to [16] an equivalent nullor network N is formed. Let us assume that there are $m+n+1$ nodes and R nullors in N . In accordance with the algorithm described in [24], nodes, numbered from 1 to m represent network sources, nodes from $m+1$ to $m+n$ are inner nodes (all or some of them can be considered as output nodes), and the $m+n+1^{\text{th}}$ node is the common node for the nullor network. Thus:

$$n_f + n_e = N_f + N_e = R \quad (29)$$

where n_f , n_e , N_f and N_e are the number of the nullators connected between inner nodes, the number of grounded nullators, the number of the norators connected between inner nodes, and the number of grounded norators, respectively.

The sequence of the nodes in the nullor network is determined as follows:

- incoming(sources) nodes - $1, \dots, m$,
- outgoing nodes - $m + 1, \dots, m + n$, are as follows:
 - p nodes, connected to edges of passive elements,
 - N_e nodes, connected to the ground by means of a norator,
 - $2N_f$ nodes, connected to N_f norators,
 - N_{fr} nodes, connected to a norator that is situated between two nodes, one of them is connected to a nullator,
 - n_f' nodes (that is one of the two nodes) connected with the nullators,
- $R = n_f + n_e$ nodes that are removed, as follows:
 - n_f nodes, corresponding to the second node, connected to a nullator,
 - n_e nodes, connected to grounded nullators.

Accordingly, an equivalent nullor network \mathbf{N} is composed that is presented by an initial modified *Coates* flow-graph.

4.1.2. Rules for modification of the initial flow-graph

In order to reflect the nullator-norator pair influence on the network transfer functions, some transformations of the modified *Coates* flow-graph are performed according to the algorithm described in [24]. From the notes made in Section 4.1.1., it can be seen that R vertices are removed from the initial flow-graph. They correspond to the number of nullators in the nullor network.

Rule 1: When a node k in the nullor network \mathbf{N} , shown in **Fig. 21(a)**, is connected to the common node by a norator, all incoming edges Y_{ki} , including the self-loop Y_{kk} , in vertex k in the corresponding flow-graph presented in **Fig. 21(b)**, are removed, *i.e.* $Y_{ki} = 0$ and $Y_{kk} = 0$. The equivalent flow-graph is shown in **Fig. 21(c)**.

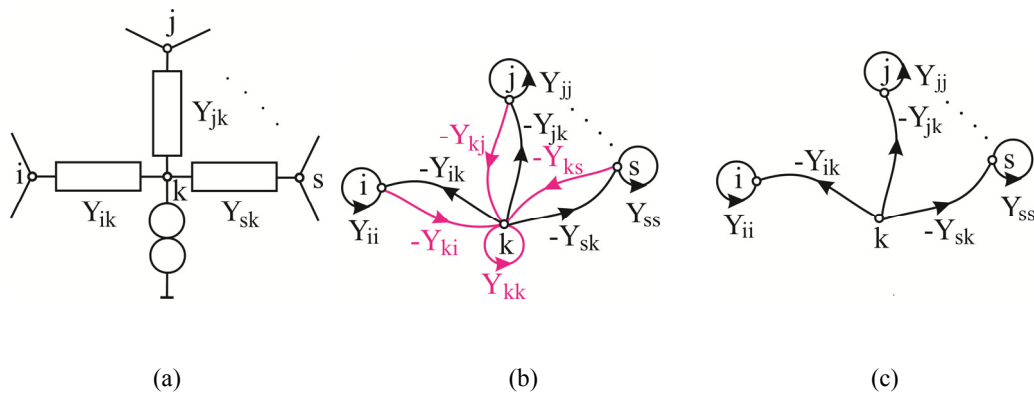


Figure 21: Rule 1 of modification of the initial flow-graph.

Rule 2: When a node k in the nullor network \mathbf{N} shown in **Fig. 22(a)**, is connected with the common node by a nullator, all outgoing edges Y_{ik} , including the self-loop Y_{kk} , in vertex k in the corresponding flow-graph presented in **Fig. 22(b)**, are removed, i.e. $Y_{ik} = 0$ and $Y_{kk} = 0$. The equivalent flow-graph is shown in **Fig. 22(c)**.

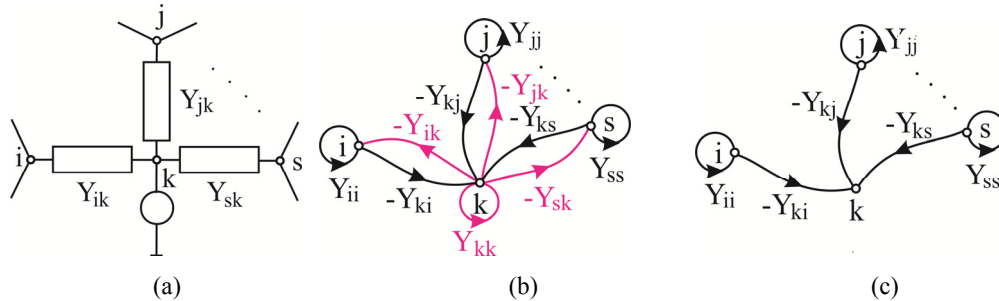


Figure 22: Rule 2 of modification of the initial flow-graph.

Rule 3: When a nullator is connected between a pair of nodes k and l in the nullor network \mathbf{N} (see **Fig. 23(a)**) all outgoing edges from vertex k in the corresponding flow-graph presented in **Fig. 23(b)**, are removed and the originals of the edges Y_{kk} and Y_{ik} are moved toward vertex l . The equivalent flow-graph is shown in **Fig. 23(c)**.

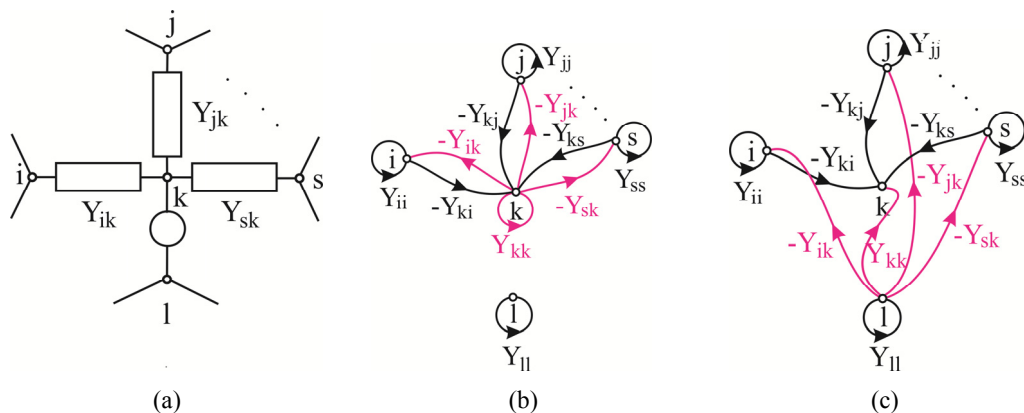


Figure 23: Rule 3 of modification of the initial flow-graph.

Rule 4: When a norator is connected between a pair of nodes k and l in the nullor network \mathbf{N} , shown in **Fig. 24(a)**, the ends of all incoming edges Y_{ki} into vertex k , including the self-loop Y_{kk} , in the corresponding flow-graph presented in **Fig. 24(b)**, are moved to vertex l , for $l \neq k$. The equivalent flow-graph is shown in **Fig. 24(c)**.

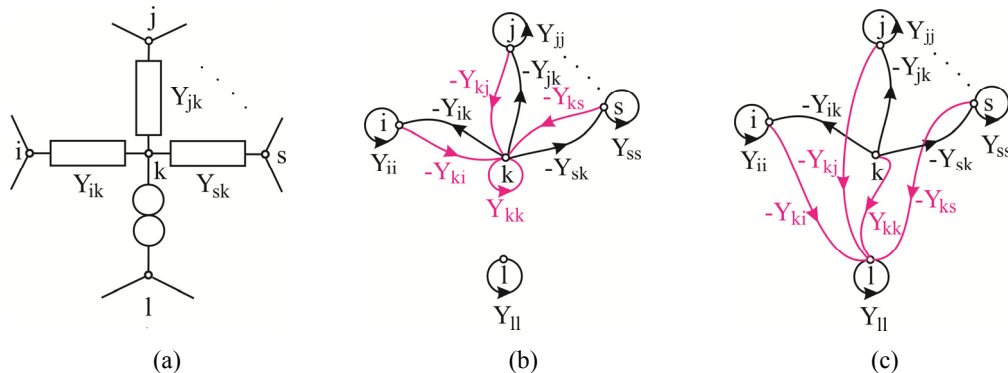


Figure 24: Rule 4 of modification of the initial flow-graph.

Rule 5: Control for all l_i vertices, corresponding to $R = n_f + n_e$ nodes in the nullor network, respectively to R vertices in the flow-graph. These vertices are the $(n+1)^{\text{th}}$ and $(n+2)^{\text{nd}}$ vertex shown in **Fig. 25(a)**. Each outgoing vertex has to have at least one incoming and one outgoing edge, and in this case, the vertex is a determined one. This requirement is not performed for the $(n-1)^{\text{th}}$ vertex in **Fig. 25(a)**. Then l_i^{th} vertex has to be united to the undetermined outgoing vertex, if it is available. At both cases, the ends of the incoming edges into the l_i^{th} vertex, are moved toward the undetermined outgoing $(n-1)^{\text{th}}$ vertex or the next determined outgoing n^{th} vertex (see **Fig. 25(b)**).

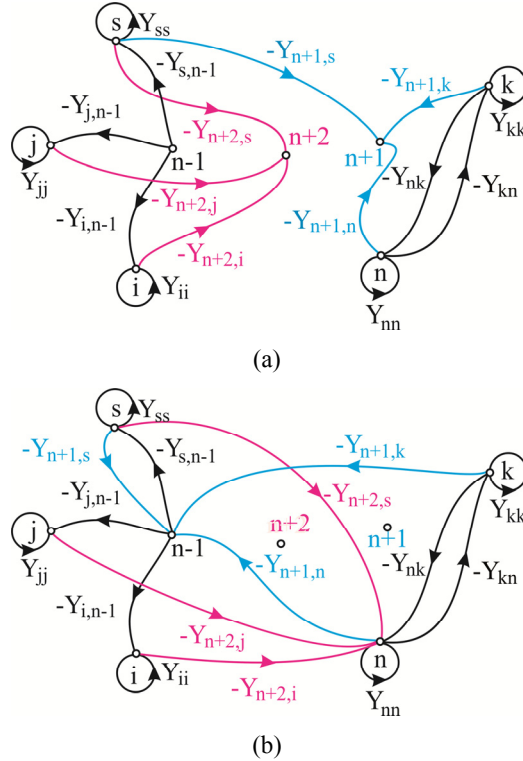


Figure 25: Rule 5 of modification of the initial flow-graph.

4.1.3. Determination of first-order symbolic transfer function sensitivity with respect to a parameter $Y(s)$ of a passive element

We suppose that the voltage transfer function $T_{kq}(s)$ is under consideration. From a practical point of view it is necessary to find first-order transfer function sensitivity $S_{Y(s)}^{T_{kq}(s)}$ with respect to the value of some passive network element $Y(s)$.

$$S_{Y(s)}^{T_{kq}(s)} = \frac{Y(s)}{T_{kq}(s)} \frac{\partial T_{kq}(s)}{\partial Y(s)} = \frac{Y(s)}{T_{kq}(s)} \sum_{j,i} \frac{\partial T_{kq}(s)}{\partial Y_{ji}(s)} \frac{dY_{ji}(s)}{dY(s)} \quad (30)$$

where $Y_{ji}(s) = a_{ji}(s) + Y(s)$ is an element of the matrix $\mathbf{Y}(s)$ reduced according to Chapter 3, Section 6; and $a_{ji}(s)$ contains other network parameters, for $i = 1, \dots, m+n$; $j = m+1, \dots, m+n$.

According to [23] the derivative $\partial T_{kq}(s)/\partial Y_{ji}(s)$ is given as follows:

$$\partial T_{kq}(s)/\partial Y_{ji}(s) = T_{iq}(s)T_{kj}(s) \quad (31)$$

The partial transfer functions $T_{iq}(s)$ and $T_{kj}(s)$ can be obtained by the modified *Coates* flow-graph \mathbf{G}^{MC} . For this reason, three sub-graphs follow, they are illustrated in Section 4.2.:

- \mathbf{G}_0^{MC} is obtained from \mathbf{G}^{MC} due to the removal of all outgoing edges from the vertex-source,
- \mathbf{G}_{k1}^{MC} , for $k = 2, \dots, n$, is obtained from \mathbf{G}^{MC} due to the removal of all outgoing edges, including the self-loop in the vertex k with a signal $V_k(s)$ and moving the vertex-source into the vertex k . As a result $Y_{jk} = 0$, $Y_{kk} = 0$ and the originals of the outgoing edges from the vertex-source are moved toward the vertex k ,
- \mathbf{G}_{kj}^{MC} is obtained from \mathbf{G}_0^{MC} by removing all outgoing edges, including the self-loop from vertex k , as well as by removing all incoming edges including the self-loop, from vertex j and must be added an edge $Y_{jk} = -1$.

Thus the transfer function $T_{kq}(s)$ is:

$$T_{kq}(s) = \frac{\sum_{Q=1}^R (-1)^{N_Q} P_Q}{\sum_{K=1}^L (-1)^{N_K} P_K} = \frac{\Delta_{kq}}{\Delta} \quad (32)$$

where Q is the separation of loops of \mathbf{G}_{kq}^{MC} ,

K - the separation of loops of \mathbf{G}_0^{MC} ,

N_Q - the number of the loops in the Q^{th} separation of loops of \mathbf{G}_{kq}^{MC} ,

N_K - the number of the loops in the K^{th} separation of loops of \mathbf{G}_0^{MC} ,

R - the number of the one factors in \mathbf{G}_{kq}^{MC} ,

L - the number of the one factors in \mathbf{G}_0^{MC} ,

P_Q - the product of the loop transmission coefficients in Q^{th} separation of loops of \mathbf{G}_{kq}^{MC} ,

P_K - the product of the loop transmission coefficients in K^{th} separation of loops of \mathbf{G}_0^{MC} .

Once the nullor network is established, a modification of the modified *Coates* flow-graph is done by the rules described in Section 4.1.2. A computer program "HoneySen" was proposed in [23]. It was developed using the modified *Coates* flow-graph theory. It automatically generates symbolic transfer functions and first-order symbolic transfer function sensitivity with respect to a passive element.

4.2. Application examples

Example 1 A second-order low-pass filter is shown in **Fig. 26**. The problem consist of determining the voltage transfer function $T(s) = U_o(s)/U_i(s) = U_3/U_1$ and its sensitivity with respect to G_2 .

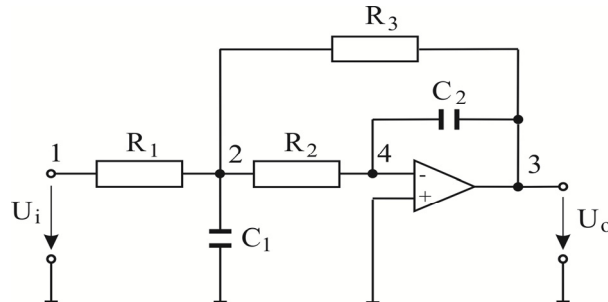


Figure 26: Second-order low-pass filter.

According to Chapter 3, Section 4 and Section 4.1.1., an equivalent nullor network **N**, which is shown in **Fig. 27**, is composed. An initial form of the modified *Coates* flow-graph for the passive part of the network is given in **Fig. 28**. By applying the sequence of modifications, which were detailed in Section 4.1.2. (see **Fig. 29**), the modified *Coates* flow-graph is obtained. It is presented in **Fig. 30**.

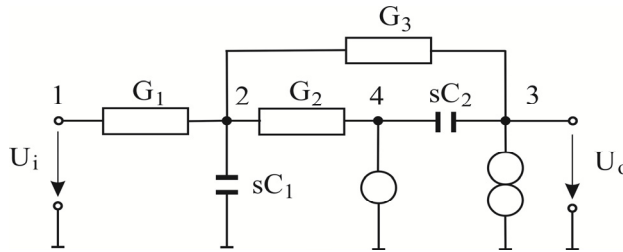


Figure 27: Equivalent nullor network.

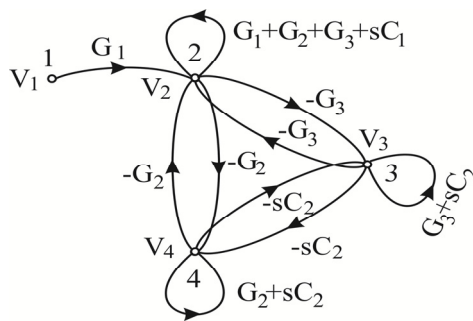


Figure 28: Initial modified *Coates* flow-graph.

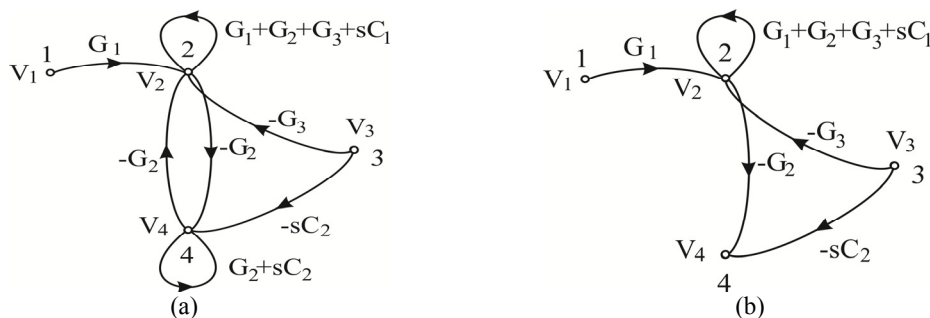


Figure 29: Modification of the initial flow-graph.

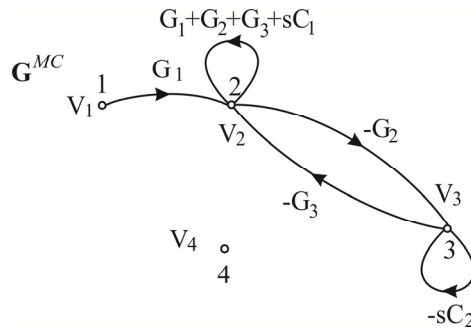


Figure 30: Modified Coates flow-graph \mathbf{G}^{MC} .

Then taking into account (30) and (32), the sensitivity $S_{G_2}^{T_{31}}$ is:

$$S_{G_2}^{T_{31}} = \frac{G_2}{T_{31}} \frac{\partial T_{31}}{\partial G_2} = \frac{G_2}{T_{31}} \left(T_{21} T_{32} \frac{dY_{22}}{dG_2} + T_{21} T_{33} \frac{dY_{32}}{dG_2} \right) \quad (33)$$

The partial transfer functions T_{31} , T_{21} , T_{32} and T_{33} are obtained by modified Coates flow-graph \mathbf{G}^{MC} , using sub-graphs G_{31}^{MC} , G_{21}^{MC} , G_{32}^{MC} , G_{33}^{MC} , and G_0^{MC} , respectively.

- Sub-graph G_0^{MC} that is shown in Fig. 31(a), is obtained from \mathbf{G}^{MC} by removing all outgoing edges from the vertex-source. Figs. 31(b, c) show the sub-graph's 1Fs.

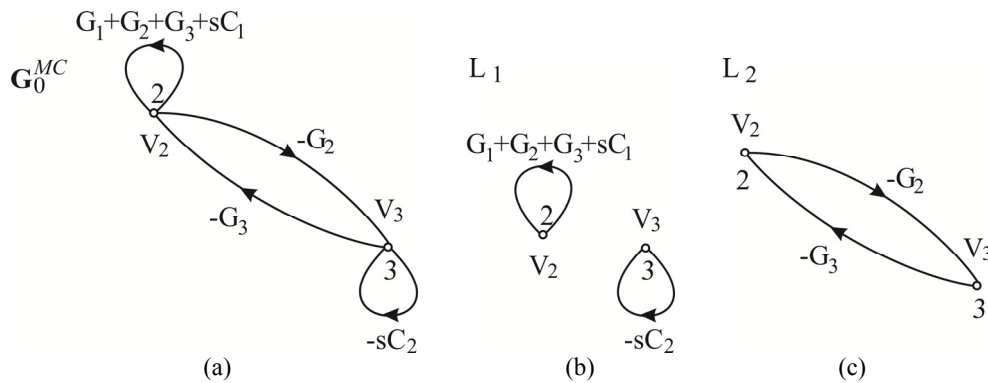


Figure 31: Sub-graph G_0^{MC} and its 1Fs.

Thus $L = 2$, $N_1 = 2$, $N_2 = 1$ and

$$\sum_{K=1}^L (-1)^{N_k} P_K = \Delta = (-1)^2 (G_1 + G_2 + G_3 + sC_1)(-sC_2) + (-1)^1 (-G_2)(-G_3) = - (G_1 + G_2 + G_3 + sC_1)sC_2 - G_2G_3 \quad (34)$$

- Sub-graph G_{31}^{MC} , which is shown in Fig. 32(a), is obtained from \mathbf{G}^{MC} by removing all outgoing edges, including the self-loop in vertex 3 with a signal $V_3(s)$, and by moving the vertex-source into vertex 3. Fig. 32(b) shows the sub-graph's 1F.

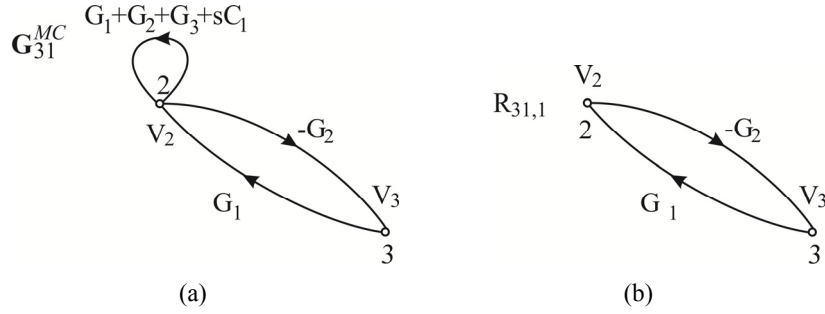


Figure 32: Sub-graph G_{31}^{MC} and its 1F.

Then, for $R = 1$ and $N_1 = 1$ we obtain:

$$\sum_{Q=1}^R (-1)^{N_Q} P_Q = \Delta_{31} = (-1)^1 G_1 (-G_2) = G_1 G_2 \quad (35)$$

Taking into account (32), the transfer function $T_{31}(s)$ is expressed as follows:

$$T(s) = T_{31}(s) = \frac{\sum_{Q=1}^R (-1)^{N_Q} P_Q}{\sum_{K=1}^L (-1)^{N_K} P_K} = \frac{\Delta_{31}}{\Delta} = \frac{G_1 G_2}{-(G_1 + G_2 + G_3 + sC_1)sC_2 - G_2 G_3} \quad (36)$$

- Sub-graph G_{21}^{MC} that is shown in **Fig. 33(a)**, is obtained from G^{MC} by removing all outgoing edges, including the self-loop in vertex 2 with a signal $V_2(s)$, and moving the vertex-source into vertex 2. **Fig. 33(b)** shows the sub-graph's 1F.

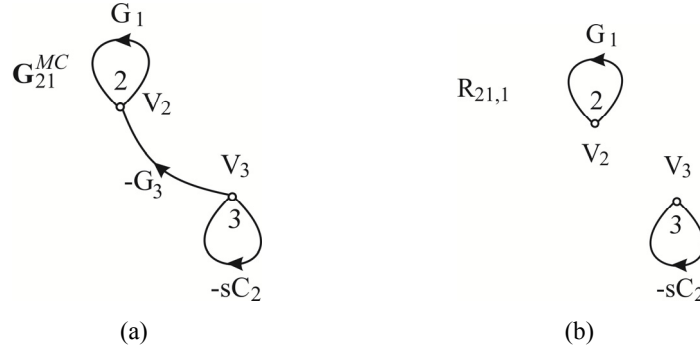


Figure 33: Sub-graph G_{21}^{MC} and its 1F.

For $R = 1$ and $N_1 = 2$, and taking into account (32), the transfer function $T_{21}(s)$ can be expressed as follows:

$$\sum_{Q=1}^R (-1)^{N_Q} P_Q = \Delta_{21} = (-1)^2 G_1 (-sC_2) = -G_1 sC_2 \quad (37)$$

$$T_{21}(s) = \frac{\sum_{Q=1}^R (-1)^{N_Q} P_Q}{\sum_{K=1}^L (-1)^{N_K} P_K} = \frac{\Delta_{21}}{\Delta} = \frac{G_1 sC_2}{(G_1 + G_2 + G_3 + sC_1)sC_2 + G_2 G_3} \quad (38)$$

- Sub-graph G_{32}^{MC} , which is presented in **Fig. 34(a)**, is obtained from G_0^{MC} by removing all outgoing edges, including the self-loop, from vertex 3, as well as by removing all incoming edges, including the self-loop, from the vertex 2. An edge $Y_{23} = -1$ must be added. **Fig. 34(b)** gives the sub-graph's 1F.

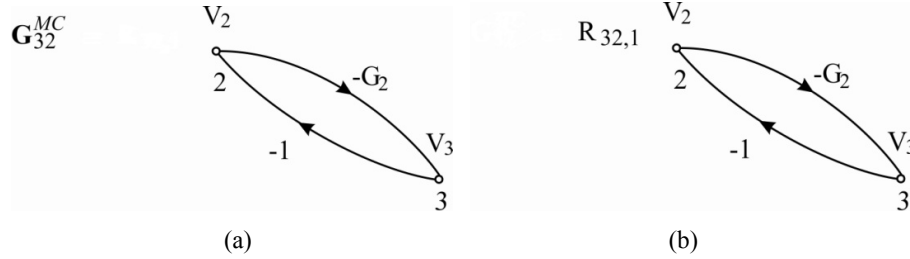


Figure 34: Sub-graph G_{32}^{MC} and its 1F.

Consequently, the transfer function $T_{32}(s)$ can be expressed as follows:

$$\sum_{Q=1}^R (-1)^{N_Q} P_Q = \Delta_{32} = (-1)^1 (-1) (-G_2) = -G_2 \quad (39)$$

$$T_{32}(s) = \frac{\sum_{Q=1}^R (-1)^{N_Q} P_Q}{\sum_{K=1}^L (-1)^{N_K} P_K} = \frac{\Delta_{32}}{\Delta} = \frac{G_2}{(G_1 + G_2 + G_3 + sC_1)sC_2 + G_2G_3} \quad (40)$$

- Sub-graph G_{33}^{MC} , which is shown in **Fig. 35(a)**, is obtained from G_0^{MC} by removing all outgoing and incoming edges, including the self-loop, from vertex 3. A n edge $Y_{33} = -1$ has to be added. **Fig. 35(b)** gives the sub-graph's 1F.

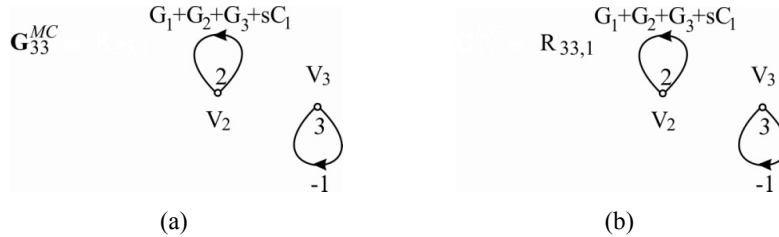


Figure 35: Sub-graph G_{33}^{MC} and its 1F.

The number of the loops in this one factor is $N_1 = 2$, then we have:

$$\sum_{Q=1}^R (-1)^{N_Q} P_Q = \Delta_{33} = (-1)^2 (G_1 + G_2 + G_3 + sC_1) (-1) = -(G_1 + G_2 + G_3 + sC_1) \quad (41)$$

The transfer function $T_{33}(s)$ is expressed as follows:

$$T_{33}(s) = \frac{\sum_{Q=1}^R (-1)^{N_Q} P_Q}{\sum_{K=1}^L (-1)^{N_K} P_K} = \frac{\Delta_{33}}{\Delta} = \frac{G_1 + G_2 + G_3 + sC_1}{(G_1 + G_2 + G_3 + sC_1)sC_2 + G_2G_3} \quad (42)$$

According to (33) the sensitivity of the transfer function is obtained. It is given by (43).

$$S_{G_2}^{T_{31}} = \frac{G_2}{G_1 G_2} \left[\frac{(-G_1 s C_2)(-G_2) - (G_1 s C_2)(G_1 + G_2 + G_3 + s C_1)}{-(G_1 + G_2 + G_3 + s C_1) s C_2 - G_2 G_3} \right] = \frac{s C_2 (G_1 + G_3 + s C_1)}{(G_1 + G_2 + G_3 + s C_1) s C_2 + G_2 G_3} \quad (43)$$

The symbolic results for the first-order transfer function sensitivity with respect to G_2 obtained by the proposed method are the same compared with the method suggested in [19]. Due to the proposed method, additional information for parameter influence upon transmission coefficients and results for the transfer functions is obtained, namely:

- all elements where the considered parameter participates are:

$$Y[2][2] = G_1 + G_2 + G_3 + s C_1 ; Y[3][2] = -G_2$$

- according to the algorithm described in [24], all transfer functions below and their products (T21 T32) ; (T21 T33) are obtained.

$$T_{31} = (G_2 G_1) / ((-(G_1 + G_2 + G_3 + s C_1) s C_2) - (G_2 G_3))$$

$$T_{21} = (-G_1 s C_2 - ((G_1 + G_2 + G_3 + s C_1) s C_2)) / ((-(G_1 + G_2 + G_3 + s C_1) s C_2) - (G_2 G_3))$$

$$T_{32} = (-G_2) / ((-(G_1 + G_2 + G_3 + s C_1) s C_2) - (G_2 G_3))$$

$$T_{33} = (-((G_1 + G_2 + G_3 + s C_1) s C_2)) / ((-(G_1 + G_2 + G_3 + s C_1) s C_2) - (G_2 G_3))$$

- the first-order transfer function sensitivity with respect to G_2 is

$$S = (G_2 / (G_2 G_1 ((-(G_1 + G_2 + G_3 + s C_1) s C_2) - (G_2 G_3)) ((-G_1 s C_2) (-G_2)) - (-G_1 s C_2) (-((G_1 + G_2 + G_3 + s C_1) s C_2))))$$

Example 2 An impedance converter is shown in **Fig. 36**. The problem consists of determining the voltage transfer function $T(s) = U_o(s)/U_i(s) = U_3/U_1$ and its sensitivity with respect to sC .

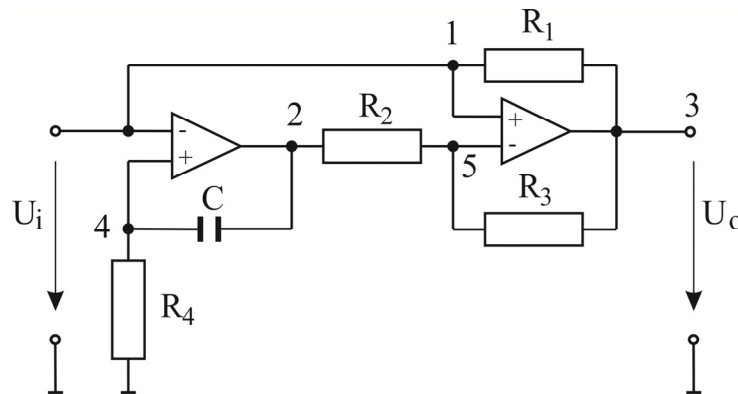


Figure 36: Impedance converter.

In accordance with Chapter 3, Section 4, and Section 4.1.1. an equivalent nullor network N is established. It is given in **Fig. 37**.

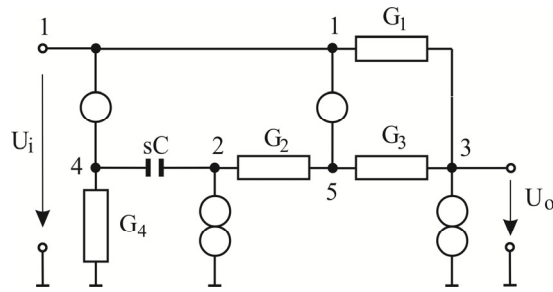


Figure 37: Equivalent nullor network.

An initial form of the modified *Coates* flow-graph for the passive part of the network is constructed. It is shown in Fig. 38. Nodes, corresponding to vertices in the original flow-graph that are removed, have numbers 4 and 5. After performing the sequence of modifications that were detailed in Section 4.1.2. (which are shown in Fig. 39), the modified *Coates* flow-graph is obtained. It is presented in Fig. 40.

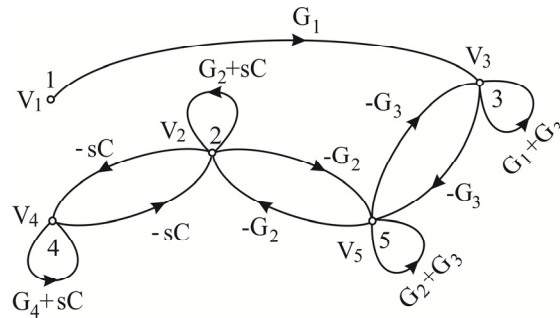


Figure 38: Initial modified *Coates* flow-graph.

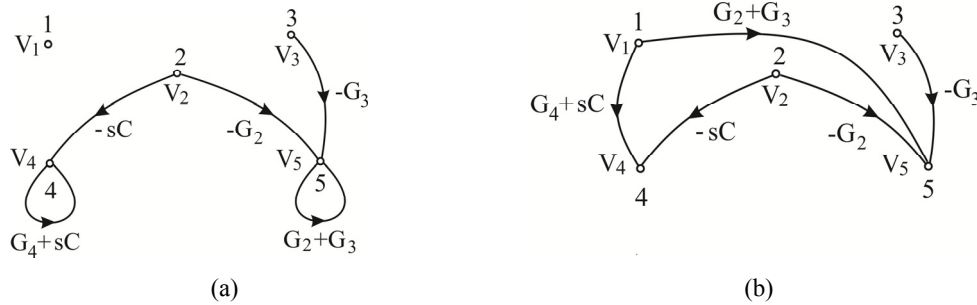


Figure 39: Modification of the initial flow-graph.

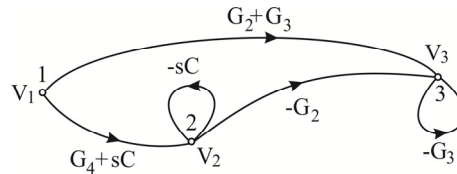


Figure 40: Modified *Coates* flow-graph \mathbf{G}^{MC} .

Taking into account (30) and (32), the sensitivity of the transfer function is expressed as follows:

$$S_{sC}^{T_{31}} = \frac{sC}{T_{31}} \frac{\partial T_{31}}{\partial sC} = \frac{sC}{T_{31}} \left(\frac{\partial T_{31}}{\partial Y_{21}} + \frac{\partial T_{31}}{\partial Y_{22}} \right) = \frac{sC}{T_{31}} \left(T_{11} T_{32} \frac{dY_{21}}{dsC} + T_{21} T_{32} \frac{dY_{22}}{dsC} \right) \quad (44)$$

The partial transfer functions T_{31} , T_{21} , T_{32} and T_{11} are obtained by the modified Coates flow-graph \mathbf{G}^{MC} , using G_{31}^{MC} , G_{21}^{MC} , G_{32}^{MC} , G_{11}^{MC} , and G_0^{MC} , respectively.

- Sub-graph G_0^{MC} that is shown in **Fig. 41(a)**, is obtained from \mathbf{G}^{MC} by removing all outgoing edges from the vertex-source. **Fig. 41(b)** gives the sub-graph's 1F.

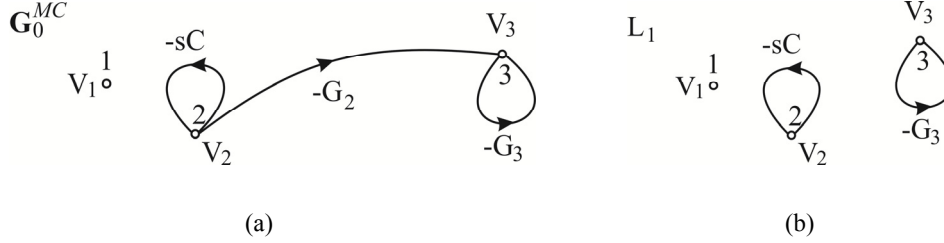


Figure 41: Sub-graph G_0^{MC} and its 1F.

Then $L = 1$, $N_1 = 2$ and

$$\sum_{K=1}^L (-1)^{N_K} P_K = \Delta = (-1)^2 (-G_3)(-sC) = G_3 sC \quad (45)$$

- Sub-graph G_{31}^{MC} , which is shown in **Fig. 42(a)**, is obtained from \mathbf{G}^{MC} by removing all outgoing edges, including the self-loop in vertex 3 with a signal $V_3(s)$, and moving the vertex-source into vertex 3. **Figs. 42(b, c)** give the sub-graph's 1Fs.

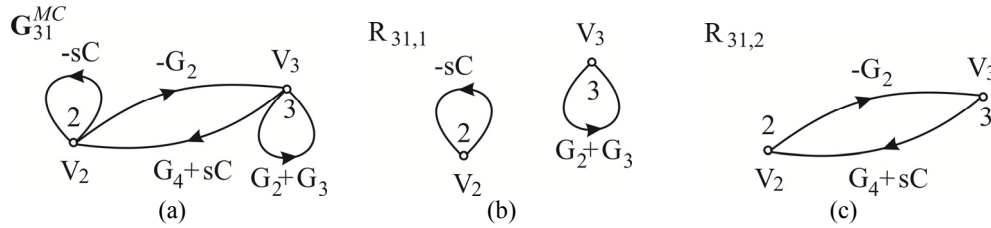


Figure 42: Sub-graph G_{31}^{MC} and its 1Fs.

Then $R = 2$, $N_1 = 2$, $N_2 = 1$ and

$$\sum_{Q=1}^R (-1)^{N_Q} P_Q = \Delta_{31} = (-1)^2 (G_2 + G_3)(-sC) + (-1)^1 (G_4 + sC)(-G_2) = G_4 G_2 - G_3 sC \quad (46)$$

Taking into account (32), the transfer function $T_{31}(s)$ is expressed as follows:

$$T_{31}(s) = \frac{\sum_{Q=1}^R (-1)^{N_Q} P_Q}{\sum_{K=1}^L (-1)^{N_K} P_K} = \frac{\Delta_{31}}{\Delta} = \frac{G_4 G_2 - G_3 sC}{G_3 sC} = \frac{G_4 G_2}{G_3 sC} - 1 \quad (47)$$

- Sub-graph G_{21}^{MC} that is shown in **Fig. 43(a)**, is obtained from \mathbf{G}^{MC} by removing all outgoing edges, including the self-loop in vertex 2 with a signal $V_2(s)$, and moving the vertex-source to vertex 2. **Fig. 43(b)** gives the sub-graph's 1F.

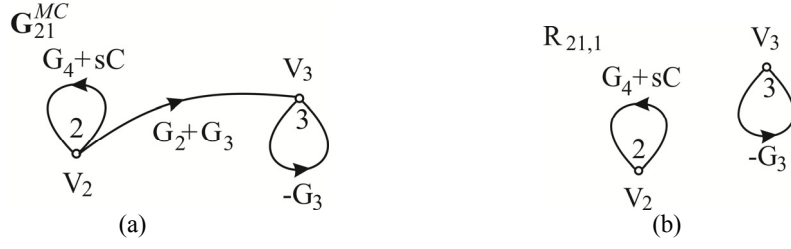


Figure 43: Sub-graph G_{21}^{MC} and its 1F.

Then, for $R = 1$ and $N_1 = 2$ we obtain,

$$\sum_{Q=1}^R (-1)^{N_Q} P_Q = \Delta_{21} = (-1)^2 (G_4 + sC)(-G_3) = -(G_4 + sC)G_3 \quad (48)$$

Taking into account (32), the transfer function $T_{21}(s)$ is:

$$T_{21}(s) = \frac{\sum_{Q=1}^R (-1)^{N_Q} P_Q}{\sum_{K=1}^L (-1)^{N_K} P_K} = \frac{\Delta_{21}}{\Delta} = \frac{-(G_4 + sC)G_3}{G_3 sC} = 1 - \frac{G_4}{sC} \quad (49)$$

- Sub-graph G_{32}^{MC} , which is shown in **Fig. 44(a)**, is obtained from G_0^{MC} by removing all outgoing edges, including the self-loop, from the vertex 3, as well as by removing all incoming edges, including the self-loop, from the vertex 2. An edge $Y_{23} = -1$ is added. **Fig. 44(b)** gives the sub-graph's 1F.

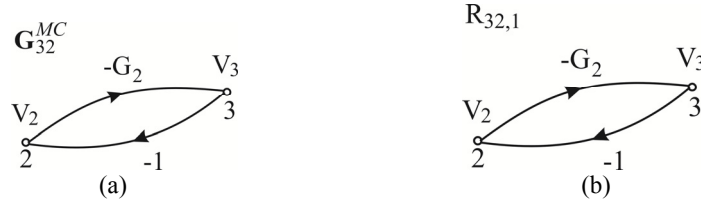


Figure 44: Sub-graph G_{32}^{MC} and its 1F.

The product of loop transmission coefficients and the transfer function T_{32} are given by expressions (50) and (51), respectively:

$$\sum_{Q=1}^R (-1)^{N_Q} P_Q = \Delta_{32} = (-1)^1 (-1)(-G_2) = -G_2 \quad (50)$$

$$T_{32}(s) = \frac{\sum_{Q=1}^R (-1)^{N_Q} P_Q}{\sum_{K=1}^L (-1)^{N_K} P_K} = \frac{\Delta_{32}}{\Delta} = \frac{-G_2}{G_3 sC} \quad (51)$$

- Sub-graph G_{11}^{MC} , which is shown in **Fig. 45(a)**, is obtained from G^{MC} by removing all outgoing and incoming edges, including the self-loop, from vertex 1. An edge $Y_{11} = -1$ is added. **Fig. 45(b)** gives the sub-graph's 1F.

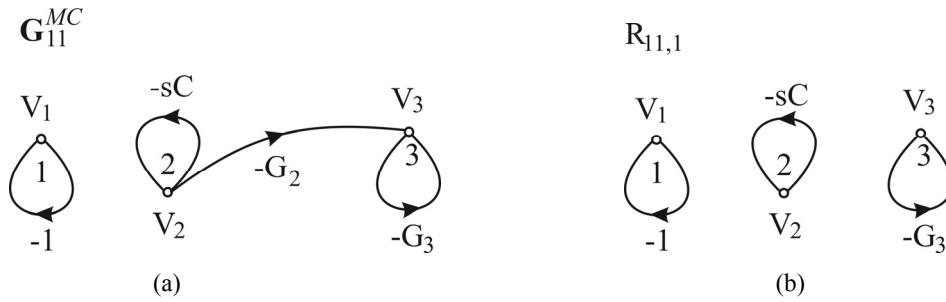


Figure 45: Sub-graph G_{11}^{MC} and its 1F.

For $R = 1$, $N = 3$, the expression follows

$$\sum_{Q=1}^R (-1)^{N_Q} P_Q = \Delta_{11} = (-1)^3 (-1)(-sC)(-G_3) = G_3 sC \tag{52}$$

Then, the transfer function T_{11} and the sensitivity $S_{sC}^{T_{31}}$ of the transfer function T_{31} with respect to the parameter sC are expressed as follow:

$$T_{11}(s) = \frac{\Delta_{11}}{\Delta} = \frac{G_3 sC}{G_3 sC} = 1 \tag{53}$$

$$S_{sC}^{T_{31}} = \frac{sC}{G_4 G_2 - G_3 sC} \left[\frac{G_3 sC (-G_2) - ((G_4 + sC) G_3) (-G_2)}{G_3 sC} \right] = \frac{G_2 G_4 + 2G_2 sC}{G_3 sC - G_2 G_4} \tag{54}$$

Finally, the symbolic result for the first-order transfer function sensitivity with respect to sC is:

$$S = (sC / (((-sC (G2+G3)))+(G2 (G4+sC)) sC G3)) / ((sC G3 (-G2))-(((G4+sC) G3) (-G2))).$$

Example 3 A second-order high-pass filter is shown in **Fig. 46**. The problem consists of determining the voltage transfer function $T(s) = U_o(s)/U_i(s) = U_3/U_1$ and its sensitivity with respect to sC_2 .

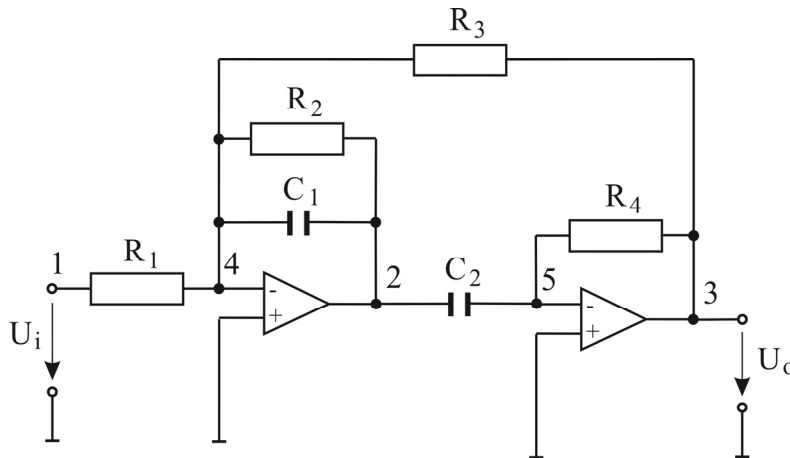


Figure 46: Second-order high-pass filter.

According to Chapter 3, Section 4, and Section 4.1.1, an equivalent nullor network **N**, which is shown in **Fig. 47**, is constructed. An initial form of the modified *Coates* flow graph for the passive part of the network is shown in **Fig. 48**.

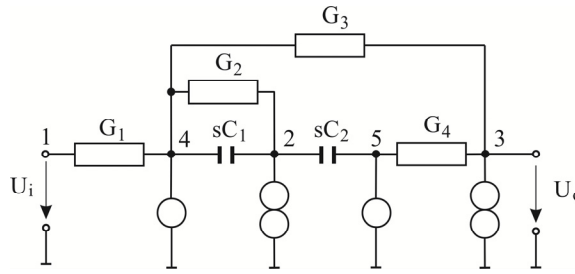


Figure 47: Equivalent nullor network.

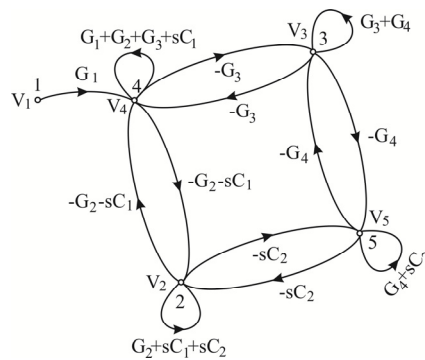


Figure 48: Initial modified *Coates* flow-graph.

By applying the sequence of modification given in Section 4.1.2., and shown in **Fig. 49**, the modified *Coates* flow-graph is obtained. It is presented in **Fig. 50**.

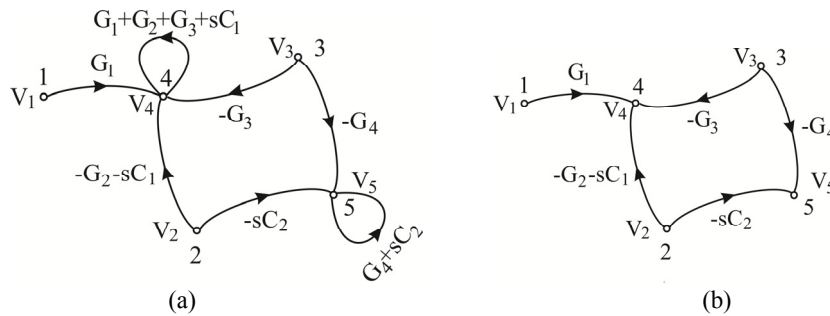


Figure 49: Modification of the initial flow-graph.

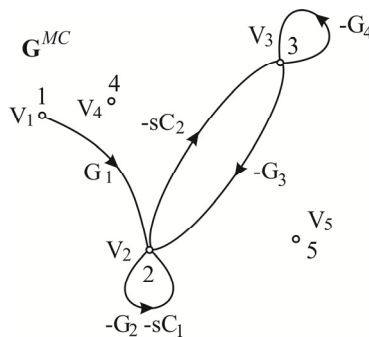


Figure 50: Modified *Coates* flow-graph.

Then, taking into account (30) and (32) the sensitivity of the transfer function is expressed as follows:

$$S_{sC_2}^{T_{31}} = \frac{sC_2}{T_{31}} \frac{\partial T_{31}}{\partial sC_2} = \frac{sC_2}{T_{31}} T_{21} T_{33} \frac{dY_{32}}{dsC_2} \quad (55)$$

The partial transfer functions T_{31} , T_{21} and T_{33} are obtained by the modified Coates flow-graph \mathbf{G}^{MC} , using sub-graphs G_{31}^{MC} , G_{21}^{MC} , G_{33}^{MC} , and G_0^{MC} , respectively:

- Sub-graph G_0^{MC} , which is shown in **Fig. 51(a)**, is obtained from \mathbf{G}^{MC} by removing all outgoing edges from the vertex-source. **Figs. 51(b, c)** give the sub-graph's 1Fs.

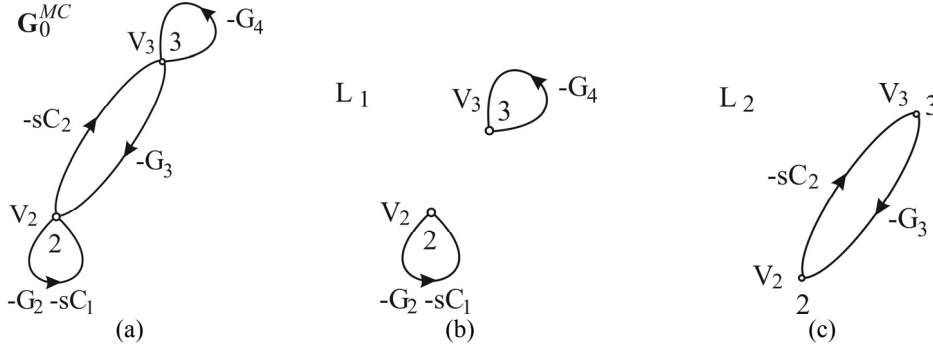


Figure 51: Sub-graph G_0^{MC} and its 1Fs.

Thus $L = 2$, $N_1 = 2$, $N_2 = 1$ and

$$\begin{aligned} \sum_{K=1}^L (-1)^{N_k} P_K &= \Delta = (-1)^2 (-G_2 - sC_1)(-G_4) + (-1)^1 (-G_3)(-sC_2) = \\ &= G_2 G_4 + G_4 sC_1 - G_3 sC_2 \end{aligned} \quad (56)$$

- Sub-graph G_{31}^{MC} , which is shown in **Fig. 52(a)**, is obtained from \mathbf{G}^{MC} by removing all outgoing edges, including the self-loop in vertex 3 with a signal $V_3(s)$, and moving the vertex-source into vertex 3. **Fig. 52(b)** presents the sub-graph's 1F.

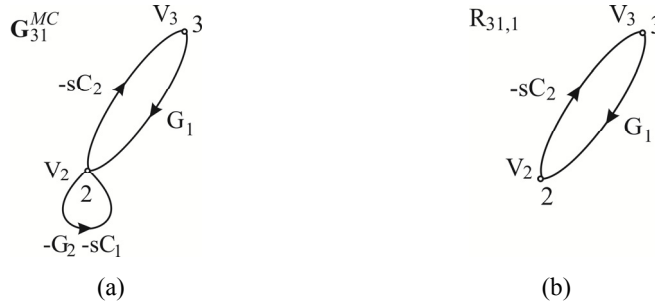


Figure 52: Sub-graph G_{31}^{MC} and its 1F.

Thus, $R = 1$, $N_1 = 1$. Taking into account (32), the transfer function $T_{31}(s)$ is expressed as follows:

$$\sum_{Q=1}^R (-1)^{N_Q} P_Q = \Delta_{31} = (-1)^1 G_1 (-sC_2) = G_1 sC_2 \quad (57)$$

$$T(s) = T_{31}(s) = \frac{\sum_{Q=1}^R (-1)^{N_Q} P_Q}{\sum_{K=1}^L (-1)^{N_K} P_K} = \frac{\Delta_{31}}{\Delta} = \frac{G_1 s C_2}{G_2 G_4 + G_4 s C_1 - G_3 s C_2} \quad (58)$$

- Sub-graph G_{21}^{MC} , which is shown in **Fig. 53(a)**, is obtained from \mathbf{G}^{MC} by removing all outgoing edges, including the self-loop in vertex 2 with a signal $V_2(s)$, and moving the vertex-source to vertex 2. **Fig. 53(b)** presents the sub-graph's 1F.

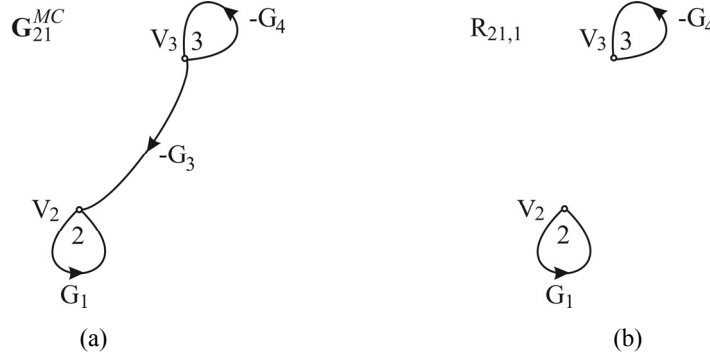


Figure 53: Sub-graph G_{21}^{MC} and its 1F.

The product of the loop transmission coefficients in this separation of loops $R_{21,1}$ is:

$$\sum_{Q=1}^R (-1)^{N_Q} P_Q = \Delta_{21} = (-1)^2 G_1 (-G_4) = -G_1 G_4 \quad (59)$$

Taking into account (32), the transfer function $T_{21}(s)$ is expressed as follows:

$$T_{21}(s) = \frac{\sum_{Q=1}^R (-1)^{N_Q} P_Q}{\sum_{K=1}^L (-1)^{N_K} P_K} = \frac{\Delta_{21}}{\Delta} = \frac{G_1 G_4}{G_2 G_4 + G_4 s C_1 - G_3 s C_2} \quad (60)$$

- Sub-graph G_{33}^{MC} that is presented in **Fig. 54(a)**, is obtained from \mathbf{G}_0^{MK} by removing all outgoing and incoming edges, including the self-loop, from vertex 3. An edge $Y_{33} = -1$ is added. **Fig. 54(b)** gives the sub-graph's 1F.

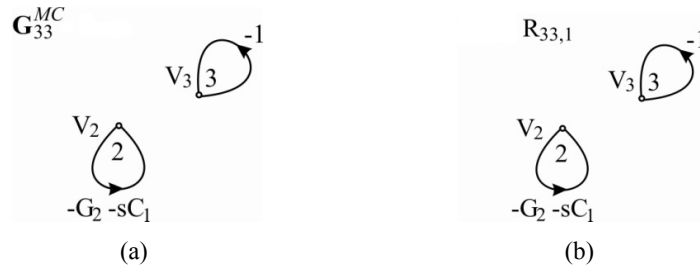


Figure 54: Sub-graph G_{33}^{MC} and its 1F.

For $R = 1$ and $N_1 = 2$ the transfer function $T_{33}(s)$ is:

$$\sum_{Q=1}^R (-1)^{N_Q} P_Q = \Delta_{33} = (-1)^2 (-G_2 - sC_1)(-1) = G_2 + sC_1 \quad (61)$$

$$T_{33}(s) = \frac{\sum_{Q=1}^R (-1)^{N_Q} P_Q}{\sum_{K=1}^L (-1)^{N_K} P_K} = \frac{\Delta_{33}}{\Delta} = \frac{G_2 + sC_1}{G_2 G_4 + G_4 sC_1 - G_3 sC_2} \quad (62)$$

Finally, according to (33) the sensitivity $S_{sC_2}^{T_{31}}$ of the transfer function $T_{31}(s)$ with respect to the parameter sC_2 is:

$$S_{sC_2}^{T_{31}} = \frac{G_4(G_2 + sC_1)}{G_3 sC_2 - G_2 G_4 - G_4 sC_1} \quad (63)$$

References

- [1] P. M. Lin, *Symbolic Network Analysis*. Elsevier, 1991.
- [2] S. J. Mason, "Feedback theory-some properties of signal flow graphs", *IRE Transactions on circuit theory*, vol. 41, no. 9, pp. 1144-1156, September 1953.
- [3] C. L. Coates, "Flow-graph solutions of linear algebraic equations", *IRE Transactions on circuit theory*, vol. CT-6, pp. 170-187, June 1959.
- [4] J. Bang-Jensen and G. Gutin, *Digraphs theory, algorithms and applications*. Springer-Verlag, 2007.
- [5] W. A. Chen, *Graph theory and its engineering applications*, World scientific publishing Co. Pte. Ltd, 1997.
- [6] D. Biolek and V. Biolková, "MC flow graphs with hybrid nodes", in *IEEE Asia-Pacific Conference on Circuits and Systems*, 2002.
- [7] J. A. Starzyk and A. Konczykowska, "Flow-graph analysis of large electronic networks", *IEEE Transactions on Circuits and Systems*, vol. CAS-33, no. 3, pp. 302-315, March 1986.
- [8] E. Tlelo-Cuautle, D. Moro-Frias, and M. Fakhfakh, Systematic design of CCI(II)(III)s by combining UGCs, in *International Design and Test Workshop*, 2008.
- [9] E. Tlelo-Cuautle, M. A. Duarte-Villaseñor, *Evolutionary electronics: automatic synthesis of analog circuits by GAs*, in *Success in Evolutionary Computation*, Series, vol. 92, Chapter 8, pp. 165-188, Yang, Ang; Shan, Yin; Bui, Lam Thu (Eds.), Springer-Verlag, Berlin, March 2008.
- [10] C.-M. Chang, "Multifunction biquadratic filters using current conveyors", *IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing*, vol. 44, no. 11, pp. 956-958, November 1997.
- [11] H. F. A. Hamed and A. A. M. Khalaf, "Differential voltage current conveyor and fully differential current conveyor in Standard CMOS technology for low voltage analog circuits applications", in *IEEE International Conference on Electronics, Circuits and Systems*, 2005.
- [12] A. M. Soliman, "Low voltage wide range CMOS differential voltage current conveyor and its applications", *Contemporary Engineering Sciences*, vol. 1, no. 3, pp. 105-126, 2008.
- [13] M. Fakhfakh and M. Pierzchala, "Computing symbolic transfer functions of CC-based circuits using Coates flow-graph", in *The International Conference on Design & Technology of Integrated Systems in Nanoscale Era*, 2010.
- [14] H.-P. Chen and Y.-Z. Liao, "High-input and low-output impedance voltage-mode universal biquadratic filter using FDCCII's", in *The International Conference on Solid-State and Integrated-Circuit Technology*, 2008.
- [15] M. Fakhfakh and M. Loulou, "Live Demonstration: CASCADES.1: A flow-graph-based symbolic analyzer", in *IEEE International Symposium on Circuits and Systems*, 2010.
- [16] D. Haigh, T. J. W. Clarke, and P. M. Radmore, "Symbolic framework for linear active circuits based on port equivalence using limit variables", *IEEE Transactions on Circuits and Systems*, vol. 53, no. 9, pp. 2011-2024, 2006.
- [17] L. T. Bruton, *RC - Active circuits theory and design*. Prentice-Hall International, Inc, London, 1980.

- [18] E. Tlelo-Cuautle, C. Sánchez-López, and F. Sandoval-Ibarra, Computing symbolic expressions in analog circuits using nullors, *Computación y Sistemas*, vol. 9, no. 2, pp. 119-132, 2005.
- [19] A. C. Davies, “Nullator – Norator equivalent networks for controlled sources, *Proceedings of the IEEE*, vol. 55, no. 5, 1967, pp.722-723.
- [20] D. Haigh, “Systematic synthesis of operational amplifier circuits by admittance matrix expansion”, in *European Conference on Circuit Theory and Design*, vol. II, 2005, pp. 115-118.
- [21] D. Haigh, T. J. W. Clarke, and P. M. Radmore, “A mathematical framework for active circuits based on port equivalence using limit variables”, in *IEEE International Symposium on Circuits and Systems*, 2006, pp. 2949-2952.
- [22] C. L. Coates, “General topological formulas for linear networks”, *IRE Transactions on Circuit Theory*, vol. CT-5. 1958.
- [23] I. N. Asenova, “Topological analysis of electrical circuit sensitivity”, PhD thesis, Higher School of Transport Engineering, Sofia, Bulgaria, 2008.
- [24] I. N. Asenova, “Method of determination of first – order symbol sensitivity by using of nullor model and modified Coates signal flow graph”, in *The International Conference ‘Elektro’*, Zilina, Slovak Republic, 2008, pp. 21-24.
- [25] S. W. Director and R. A. Rohrer, “The Generalized Adjoin Network and Sensitivity Analysis”, *IEEE Transactions on Circuit Theory*, CT-16, 1969, pp 318-323.

CHAPTER 6

Analysis and Synthesis of Electronic Circuits by the Two-Graph Method

Marian Pierzchała^{1,*} and Benedykt Rodanski²

¹Institute of Telecommunication, Teleinformatics and Acoustics, Wrocław University of Technology, Wrocław, Poland and ²School of Electrical, Mechanical and Mechatronic Systems, Faculty of Engineering and Information Technology, University of Technology, Sydney (UTS), Sydney, Australia

Abstract: The general two-graph framework of a fully symbolic and semi-symbolic analysis environment for linear, time-invariant circuits is presented. A classical two-graph approach for RLC- g_m circuits as well as its extension for circuits containing non-admittance elements is discussed. A brief introduction to approximate symbolic analysis, using the two-graph method, is included. In this chapter we also present a method of synthesis of active RC circuits on the basis of the two-graph method. The nullor approach is used to synthesize the RC network which has the voltage and the current graphs equivalent to the two-graph of the prototype LC network.

Keywords: Symbolic analysis, two-graph method, symbolic network functions, synthesis of RC active circuits, circuits with nullors, current conveyors, floating nullors, RC active filters with losses, loop matrix, cutset matrix, product matrix.

1. Introduction

This chapter has four main parts. In the first part (Sections 2-4), we consider the connected, linear, time-invariant networks with immittances, independent and controlled sources. The main purpose of this part is to introduce some basic concepts of graph theory, as applied to circuit analysis and to present a classical two-graph approach to obtaining fully symbolic expressions for relationships between selected output and input variables (voltages and currents). The two-graph method was first introduced as an effective tool to obtain symbolic determinant of the Node Admittance Matrix (NAM) of an active circuit [1]. It was therefore suitable for a limited class of circuits, namely the RLC- g_m circuits. The modified nodal analysis [2] removed this limitation, allowing analysis of circuits containing other circuit elements, such as all controlled sources, transformers, ideal op amps, *etc.* The two-graph method can also be modified to include the non-admittance components. The price we have to pay for this convenience is the increase of circuit complexity.

All fully symbolic exact analysis methods suffer from the complexity problem: the number of terms in the formula grows exponentially or superexponentially, $O(n^n)$, with the circuit size. One solution to this serious problem is to obtain an approximate expression with only the most significant terms present. The two-graph method is very suitable for this task, because it allows the generation of terms in decreasing order of magnitude [3]. Section 5 gives a brief introduction to this application of the two graph method. (There are other approaches to approximate symbolic analysis. A good review of existing literature can be found in [4]).

Often we are interested in finding the contribution of only a few components to the overall circuit response. The semi-symbolic analysis results in a formula with mixed

*Address correspondence to Marian Pierzchała: Institute of Telecommunication, Teleinformatics and Acoustics, Wrocław University of Technology, Wrocław, Poland; E-mail: Marian.Pierzchala@pwr.wroc.pl

symbolic and numeric terms. The two-graph method is also very effective here. In Section 6 we present one such application that allows calculation of numerical coefficients at any symbol combination in a single step [5].

The fourth part of this chapter (Section 7) shows the viability of the two-graph method to synthesize the active RC filters on the basis of the prototype LC filter structures [6]. The use of the Four Terminal Floating Nullor (FTFN) facilitates a transformation process from the passive LC filter structure to the architecture of the active RC circuits.

2. Fundamentals of Network Topology

Network topology deals with those properties of lumped networks which are related to the interconnection of branches only. It is one of many subfields in graph theory. In this section, which is mainly based on [7], we shall present those concept and theorems which are essential to symbolic network analysis and synthesis of electronic circuits by the two-graph method. Readers already familiar with fundamentals of network topology may omit this section.

2.1. Directed and undirected graphs

A complete description of any lumped network must contain the following information:

1. How the branches are connected.
2. The reference directions for branch currents and voltages.
3. The branch voltage-current relationships.

One natural and simple way to depict items 1 and 2 is to draw a *directed graph* G_d associated with the given network N , according to the following rules: Replace each network node by a graph node, or *vertex*. Replace each network branch by a line segment, called the graph branch or the *edge*. Each edge has an arrow in the same direction as the assumed positive current through the corresponding network branch. This arrow also serves as the branch voltage reference: the positive voltage terminal is assumed to be the tail of the arrow. With such a unified reference scheme for both voltages and currents, the directed graph G_d contains complete information about items 1 and 2 above.

There are situations where the reference directions of the currents and voltages are of no consequence (for example, topological formulae for RLC network functions). Then all the arrows in G_d may be removed. The resultant simpler graph is called the *undirected graph* associated with the network N , and is denoted by G_n .

Fig. 1(a) shows a network N ; its associated directed graph G_d is shown in **Fig. 1(b)**.

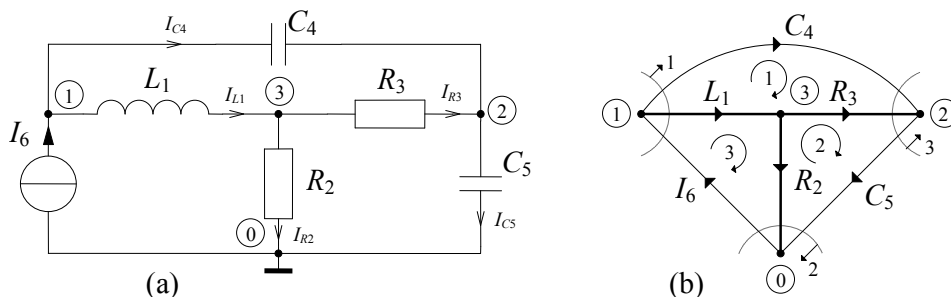


Figure 1: An electrical network (a) and its associated directed graph (b). The fundamental loops and cutsets, associated with the spanning tree (L_1, R_2, R_3) , are marked on the graph.

The following are some basic concepts in graph theory, defined in terms of an undirected graph.

Definition 1: Path. A set of branches b_1, b_2, \dots, b_m in G_n is called a path between two nodes V_j and V_k if the branches can be ordered such that:

1. Consecutive branches b_i and b_{i+1} have a common endpoint;
2. No node is the endpoint of more than two branches in the set;
3. V_j is the end point of exactly one branch in the set, and so is V_k .

For example, branches (L_1, R_2, C_5) in **Fig. 1(b)** form a path between nodes 1 and 2.

Definition 2: Connected graph. An undirected graph G_n is said to be connected if there exists a path between any two nodes of the graph. A network N or a directed graph G_d is said to be connected if the associated undirected graph G_n is connected.

Definition 3: Loop (Circuit). A subgraph G_s of a graph G_n , is called a loop if (1) G_s is connected and (2) every node of G_s has exactly two branches of G_s incident to it.

For example, branches (C_4, C_5, I_6) in **Fig. 1(b)** form a loop, and so the branches (L_1, R_2, C_4, C_5) .

Definition 4: Spanning Tree. A sub-graph G_s of a graph G_n is called a spanning tree if:

1. G_s is connected;
2. G_s has no loops;
3. G_s contains all nodes of G_n .

In a connected graph with n nodes, each spanning tree has $n-1$ branches. For example, branches (L_1, R_2, R_3) in **Fig. 1(b)** form a spanning tree and so do branches (R_2, C_5, I_6) . In this chapter we will only be concerned with spanning trees, so for simplicity, we will often use the term ‘tree’ in the remainder.

The edges that belong to a tree T are called *tree branches*, and the remaining edges are called *links* or *chords*. All links of a given tree T form what is called a *cotree* with respect to T .

Definition 5: Cutset. A set of branches of a connected graph G_s is said to be a cutset if:

1. The removal of the set of branches (but not their endpoints) results in a graph that is not connected;
2. After the removal of the set of branches, the restoration of any one branch will result in a connected graph.

For example, branches (L_1, C_4, I_6) in **Fig. 1(b)** form a cutset, and so do branches (R_2, R_3, C_4, I_6) .

2.2. Matrices associated with a directed graph

Out of many possible topological matrices that can be defined for a graph, we will introduce only most important ones that will be used in this chapter.

2.2.1. Incidence matrix

Let G_d be a connected, directed graph with n nodes and b branches. An $n \times b$ matrix $\mathbf{A}_a = [a_{ij}]$ is said to be the *complete* (or *augmented*) *incidence matrix* of G_d if its entries are defined as follows:

$a_{ij} = 1$ if branch j is incident to node i and the arrow is pointing away from node i ,
 $a_{ij} = -1$ if branch j is incident to node i and the arrow is pointing toward node i ,
 $a_{ij} = 0$ if branch j is not incident to node i .

For example, the directed graph G_d in **Fig. 1(b)** has the complete incidence matrix:

\mathbf{A}_a	L_1	R_2	R_3	C_4	C_5	I_6
1	1	0	0	1	0	-1
2	0	0	-1	-1	1	0
3	-1	1	1	0	0	0
0	0	-1	0	0	-1	1

Since elements of every column of \mathbf{A}_a add up to zero, any one row of the augmented incidence matrix may be deleted without loss of information. A matrix obtained from \mathbf{A}_a by deleting an arbitrary row is denoted by \mathbf{A} and simply called the *incidence matrix*.

For a chosen spanning tree T the incidence matrix can be partitioned as $\mathbf{A} = [\mathbf{A}_T \mathbf{A}_C]$, where columns of \mathbf{A}_T correspond to the tree branches and columns of \mathbf{A}_C correspond to the cotree branches. The matrix \mathbf{A}_T is unimodular (*i.e.*, its determinant is equal to ± 1).

For example, if we choose branches L_1 , R_2 and R_3 to form the tree in the graph of **Fig. 1(b)** and delete row 0 from \mathbf{A}_a , then matrix \mathbf{A}_T will be

\mathbf{A}_T	L_1	R_2	R_3
1	1	0	0
2	0	0	-1
3	-1	1	1

Its determinant is $|\mathbf{A}_T| = 1$.

2.2.2. Fundamental loop matrix

Let G_d be a connected, directed graph with n nodes and b branches. Let T be a spanning tree in G_d . Each link of the cotree forms a loop with the unique path through the tree, called the *fundamental loop* for that link with respect to the chosen tree T . The orientation of the fundamental loop is chosen to coincide with that of the link and is indicated by an arrow. There are $\mu = (b - n + 1)$ fundamental loops.

A matrix $\mathbf{B}_f = [b_{ij}]$ of order $\mu \times b$, whose entries are defined as follows:

$b_{ij} = 1$ if loop i contains branch j and the directions agree,
 $b_{ij} = -1$ if loop i contains branch j and the directions oppose,
 $b_{ij} = 0$ if loop i does not contain branch j ,

is called the *fundamental loop matrix*.

For example, if T is chosen to consist of branches (L_1, R_2, R_3) in **Fig. 1(b)**, the fundamental loop matrix is:

\mathbf{B}_f	L_1	R_2	R_3	C_4	C_5	I_6
1	-1	0	-1	1	0	0
2	0	-1	1	0	1	0
3	1	1	0	0	0	1

From the way the matrix \mathbf{B}_f is constructed, it follows that any fundamental loop matrix can be partitioned as $\mathbf{B}_f = [\mathbf{B}_T \ \mathbf{1}_\mu]$, where the columns of \mathbf{B}_T correspond to the tree branches, and the identity matrix of order μ , $\mathbf{1}_\mu$, corresponds to the links.

2.2.3. Fundamental cutset matrix

Consider the connected, directed graph G_d . Let T be a spanning tree in G_d . Each branch of T together with some (possibly none) links in the associated cotree form a cutset, called the *fundamental cutset*, for that tree branch, with respect to the chosen tree T . The reference arrow for the cutset is chosen to agree with the tree branch. For a connected graph with n nodes, there are $n-1$ tree branches, and hence there are $\rho = n - 1$ fundamental cutsets for each chosen tree T .

We define the *fundamental cutset matrix* to be an $\rho \times b$ matrix $\mathbf{Q}_f = [q_{ij}]$ whose entries are defined as follows:

$$\begin{aligned} q_{ij} &= 1 && \text{if branch } j \text{ is in the cutset } i, \text{ and their directions agree,} \\ q_{ij} &= -1 && \text{if branch } j \text{ is in the cutset } i, \text{ and their directions oppose,} \\ q_{ij} &= 0 && \text{if branch } j \text{ is not in the cutset } i. \end{aligned}$$

For example, if T in **Fig. 1(b)** is chosen to consist of branches (L_1, R_2, R_3) , then the fundamental cutset matrix is

\mathbf{Q}_f	L_1	R_2	R_3	C_4	C_5	I_6
1	1	0	0	1	0	-1
2	0	1	0	0	1	-1
3	0	0	1	1	-1	0

From the way the matrix is \mathbf{Q}_f defined, it follows that it can be partitioned as $\mathbf{Q}_f = [\mathbf{1}_\rho \ \mathbf{Q}_C]$, where the columns of the identity matrix of order ρ , $\mathbf{1}_\rho$, correspond to the tree branches and the columns of \mathbf{Q}_C correspond to the cotree (links).

3. Network Functions of a Closed System

3.1. Network functions in the frequency domain

The main purpose of symbolic analysis in the frequency domain is to obtain analytical expressions for relationships between selected output and input variables (voltages and currents). These relationships, called *s-expanded network functions*, are expressed as ratios of polynomials (rational functions) in the complex frequency variable s

$$T(s) = \frac{N(s)}{D(s)} = \frac{n_0 + n_1 s + \dots + n_n s^n}{d_0 + d_1 s + \dots + d_d s^d} \quad (1)$$

Each coefficient (n_i , d_j) may be either numerical, mixed numerical and symbolic (semi-symbolic) or fully symbolic in terms of circuit elements. In this chapter we are concerned only with the fully symbolic and semi-symbolic s -expanded network functions.

To motivate our discussion let us first introduce a problem, solution to which can be elegantly accomplished by the two-graph method. Suppose we want to find symbolic expressions for a voltage transmittance, $T_v = v_2/v_1$, an input impedance, $Z_{in} = v_1/i_1$, and a transresistance, $R_T = v_2/i_1$, of a circuit shown in **Fig. 2**.

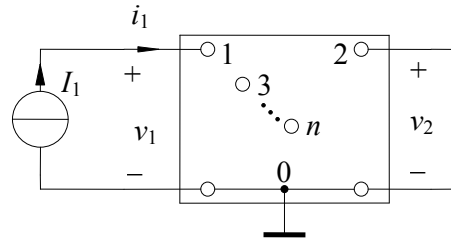


Figure 2: A circuit whose network functions are to be calculated.

Assume that the circuit can be described by the Node Admittance Matrix (NAM) or the Modified Node Admittance Matrix (MNAM), \mathbf{Y}_n . Applying nodal analysis, we can write the following equation:

$$\begin{bmatrix} I_1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1n} \\ y_{21} & y_{22} & \cdots & y_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ y_{n1} & y_{n2} & \cdots & y_{nn} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} \quad (2)$$

Now, using Cramer's rule, we can calculate voltages v_1 and v_2 as:

$$v_1 = \frac{I_1 \Delta_{11}}{\Delta}, \quad v_2 = \frac{I_1 \Delta_{12}}{\Delta} \quad (3)$$

where: $\Delta = \det(\mathbf{Y}_n)$,

Δ_{ij} is the cofactor, corresponding to the element y_{ij} of \mathbf{Y}_n .

The required network functions can now be expressed as:

$$T_v \doteq \frac{v_2}{v_1} = \frac{\Delta_{12}}{\Delta_{11}}, \quad Z_{in} \doteq \frac{v_1}{I_1} = \frac{\Delta_{11}}{\Delta}, \quad R_T \doteq \frac{v_2}{I_1} = \frac{\Delta_{12}}{\Delta} \quad (4)$$

The task of calculating symbolic expressions for Δ , Δ_{11} and Δ_{12} can be significantly simplified by the use of the following technique, sometimes called the *circuit augmentation*.

3.2. Application of a closed system

Replace the independent current source I_1 by the Voltage-Controlled Current Source (VCCS), \hat{g}_m , controlled by the output voltage v_2 and having internal admittance \hat{y} , thus forming a *closed system*, as shown in **Fig. 3**. It is important to note that symbols \hat{g}_m and \hat{y} must be unique.

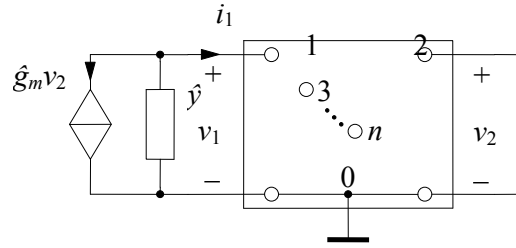


Figure 3: The closed system (augmented circuit) obtained from the circuit in **Fig. 2**.

The NAM of the augmented circuit can be written as:

$$\hat{\mathbf{Y}}_n = \begin{bmatrix} \hat{y} + y_{11} & \hat{g}_m + y_{12} & \cdots & y_{1n} \\ y_{21} & y_{22} & \cdots & y_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ y_{n1} & y_{n2} & \cdots & y_{nn} \end{bmatrix} \quad (5)$$

Calculating the determinant of (5), using Laplace expansion along the first row of $\hat{\mathbf{Y}}_n$, we obtain:

$$\hat{\Delta} = \hat{y}\Delta_{11} + y_{11}\Delta_{11} + \hat{g}_m\Delta_{12} + y_{12}\Delta_{12} + \cdots + y_{1n}\Delta_{1n} = \hat{y}\Delta_{11} + \hat{g}_m\Delta_{12} + \Delta \quad (6)$$

So, by calculating a single determinant and collecting the terms with the unique symbols \hat{g}_m and \hat{y} we have obtained both required cofactors and the determinant of the original circuit matrix.

Symbolic determinant of an arbitrary matrix may be calculated by Laplace expansion [8]. This approach will produce many cancelling terms if applied to the NAM. It is now universally recognized that the most effective way to obtain a symbolic expression for a NAM's determinant is to use the two-graph method.

4. The Two-Graph Method

4.1. Classical two-graph method

The method was originally introduced for circuits containing only passive elements and voltage-controlled current sources [1]. In this approach, for a lumped, linear, time-invariant circuit, two graphs are constructed: the voltage graph, G_V , and the current graph, G_I . Both graphs have the same number of vertices and edges, but (for active circuits) have different topology. Each vertex corresponds to a circuit node; each edge is labeled with an admittance term, corresponding to a circuit element. Assuming that the circuit is described by the node admittance matrix, the determinant of this matrix can be calculated as the sum of signed admittance products of all common spanning trees of G_V and G_I . (A common spanning tree is a set of edges that form spanning trees in both G_V and G_I .) It is important to note that no cancelling terms are generated by this approach. The sign of each product term can be determined by either a topological method or numerically, by calculating the product of the determinants of the \mathbf{A}_T matrices corresponding to the common spanning tree. If the closed system is formed, as described in Section 3, symbolic network functions can be obtained by appropriate sorting of the terms of NAM's determinant.

If the circuit contains only capacitors and (trans)conductances, the automation of the term generation process is very straightforward. The main loop generates all spanning

trees [9] in one of the two graphs, say G_I . Each tree of G_I is then checked if it is also a spanning tree of G_V . If yes, we have a common spanning tree, which is placed in an appropriate ‘bin’, depending on the presence of the unique symbol \hat{g}_m or \hat{y} (application of the sorting scheme (6)) and, if required, the number of capacitive branches (sorting on the powers of s).

If there are inductors in the circuit, each product of the common tree admittances must be multiplied by the impedance product of all inductors, and the resulting new product term must be simplified if necessary. The same process is applied if there are resistors in the circuit that must be represented in the formula by their resistance R rather than by their conductance G .

Example 1

Suppose we want to find an expression for the voltage transmittance $T_v = v_2/v_1$ for the circuit in **Fig. 1(a)**. First, we apply the circuit augmentation technique, described in section 3.2, replacing the independent current source I_6 by a VCCS $\hat{g}_m v_2$ in parallel with the admittance \hat{y} . Next, we construct the current and voltage graphs of the augmented circuit, as shown in **Fig. 4**. The edges must be labeled with the admittance terms, thus we have: $Y_1=1/Z_1=1/sL_1$, $Y_2=1/R_2=G_2$, $Y_3=1/R_3=G_3$, $Y_4=sC_4$, $Y_5=sC_5$.

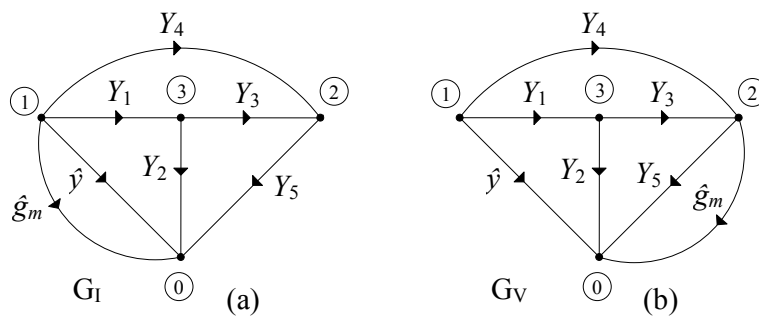


Figure 4: The current graph G_I (a) and the voltage graph G_V (b) of the closed system, obtained from the circuit in **Fig. 1(a)**.

There are 20 common spanning trees in the graphs of **Fig. 4**. The closed system determinant (6) for this circuit is

$$\begin{aligned} \hat{\Delta} = & \hat{g}_m \Delta_{12} + \hat{y} \Delta_{11} + \Delta = \hat{g}_m (Y_1 Y_3 + Y_1 Y_4 + Y_2 Y_4 + Y_3 Y_4) + \\ & \hat{y} (Y_1 Y_3 + Y_1 Y_4 + Y_1 Y_5 + Y_2 Y_3 + Y_2 Y_4 + Y_2 Y_5 + Y_3 Y_4 + Y_3 Y_5) + \\ & Y_1 Y_2 Y_3 + Y_1 Y_2 Y_4 + Y_1 Y_2 Y_5 + Y_1 Y_3 Y_5 + Y_1 Y_4 Y_5 + Y_2 Y_3 Y_4 + Y_2 Y_4 Y_5 + Y_3 Y_4 Y_5 \end{aligned} \quad (7)$$

So, according to (4), we obtain the required voltage transmittance as

$$T_v \doteq \frac{v_2}{v_1} = \frac{\Delta_{12}}{\Delta_{11}} = \frac{Y_1 Y_3 + Y_1 Y_4 + Y_2 Y_4 + Y_3 Y_4}{Y_1 Y_3 + Y_1 Y_4 + Y_1 Y_5 + Y_2 Y_3 + Y_2 Y_4 + Y_2 Y_5 + Y_3 Y_4 + Y_3 Y_5} \quad (8)$$

Now, if we wish to write the above network function as a ratio of two polynomials in s (1), the admittance symbols Y_k in (8) must be replaced by their s -expanded equivalents and the terms sorted on powers of s . First, however, since the circuit contains the inductor L_1 , we must multiply each term of the numerator and denominator by $Z_1=sL_1$ and remove the products $Y_1 Z_1$ (because they are equal to one). Finally, sorting on powers of s , we obtain the voltage transmittance in the fully s -expanded form as

$$\begin{aligned}
 T_v &= \frac{Y_3 + Y_4 + Z_1 Y_2 Y_4 + Z_1 Y_3 Y_4}{Y_3 + Y_4 + Y_5 + Z_1 Y_2 Y_3 + Z_1 Y_2 Y_4 + Z_1 Y_2 Y_5 + Z_1 Y_3 Y_4 + Z_1 Y_3 Y_5} \\
 &= \frac{G_3 + sC_4 + s^2 L_1 C_4 (G_2 + G_3)}{G_3 + s(C_4 + C_5 + L_1 G_2 G_3) + s^2 L_1 (G_2 C_4 + G_2 C_5 + G_3 C_4 + G_3 C_5)}
 \end{aligned}
 \tag{9}$$

The classical two-graph method, described above, cannot handle the non-admittance elements like the remaining three types of controlled sources, ideal transformers, *etc.* Fortunately, a simple modification will allow all types of circuit components to be included. This modification will be presented in the following section.

4.2. Modification of the two-graph method to include the non-admittance elements

It is well known that every circuit component that does not have an admittance representation (or we do not wish it to appear as an admittance in the formula) can be modeled by a circuit that contains only admittances and voltage-controlled current sources [7, 10]. Consider, for example, a current-controlled current source (CCCS), shown in Fig. 5(a). It can be modeled by a circuit containing three VCCS's, as shown in Fig. 5(b). This model of a CCCS has its representation in the current and voltage graphs, the so called two-graph 'stamp', shown in Fig. 5(c). Note that an additional circuit node, $n+1$, and two new branches were created, increasing the circuit complexity.

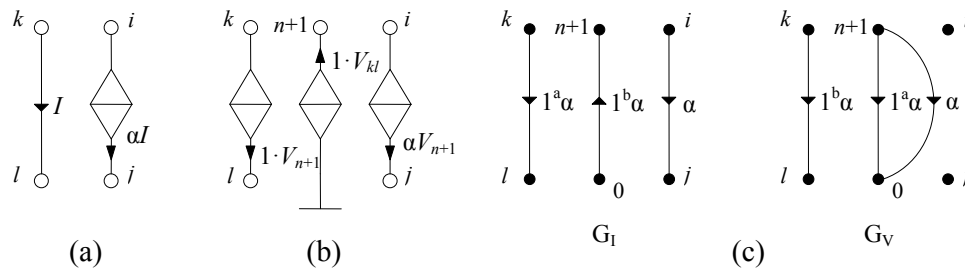


Figure 5: A current-controlled current source (a), its model (b) and the corresponding two-graph representation (c).

In Fig. 5(b) the two coupled VCCSs with their transconductances equal to one form a well known circuit component, called the *unity gyrator*. The unity gyrator(s) can be used to model all other circuit elements, including controlled sources, transformers, *etc.* This technique can also be used to model any impedance Z by an equivalent admittance Y . So, all non-admittance elements can be represented by their two-graph stamps [11]. Some of these stamps are shown in Table 1.

Two special elements are also included in Table 1: *nullator* and *norator*. A nullator is defined as having its voltage and current simultaneously equal to zero. A norator has an arbitrary voltage and current. These elements belong to the class of *singular elements* [12] and are useful in modeling the ideal behavior of real circuit components (an op amp, for example). These elements will be used in Section 7.

One question remains: how to treat the unity transconductances symbolically? Being equal to one siemens, they do not contribute (numerically) to a product of spanning tree admittances, so they can be removed from each product term as soon as a common tree is generated that contain edges corresponding to those transconductances. The only exception is a common tree consisting solely of such unity transconductances. In these cases the tree admittance product is represented by the symbol ‘1’.

Table 1: Modified Two-Graph Element Stamps for Some Circuit Elements.

Element	Symbol	Current Graph	Voltage Graph
Admittance G, sC, Y			
Impedance R, sL, Z			
VCCS			
VCVS			
CCCS			
CCVS			
Nullator			
Norator			

In computer implementation of this extension to the two-graph method it is not necessary to label the unity VCCS's in any special way, other than flag them for subsequent removal. For clarity of presentation, however, the labels '1^aX', '1^bX' are used here, where 'X' is replaced by the label of the modeled component.

Example 2

To illustrate usefulness of the modified two-graph method, consider the following simple example. **Fig. 6(a)** shows an augmented small-signal, mid-band equivalent circuit of a BJT common-emitter amplifier. It is required to find the symbolic expression for the voltage transfer function, $T_v = v_2/v_1$, of this circuit. The symbolic expression should only use the element symbols as they appear on the diagram.

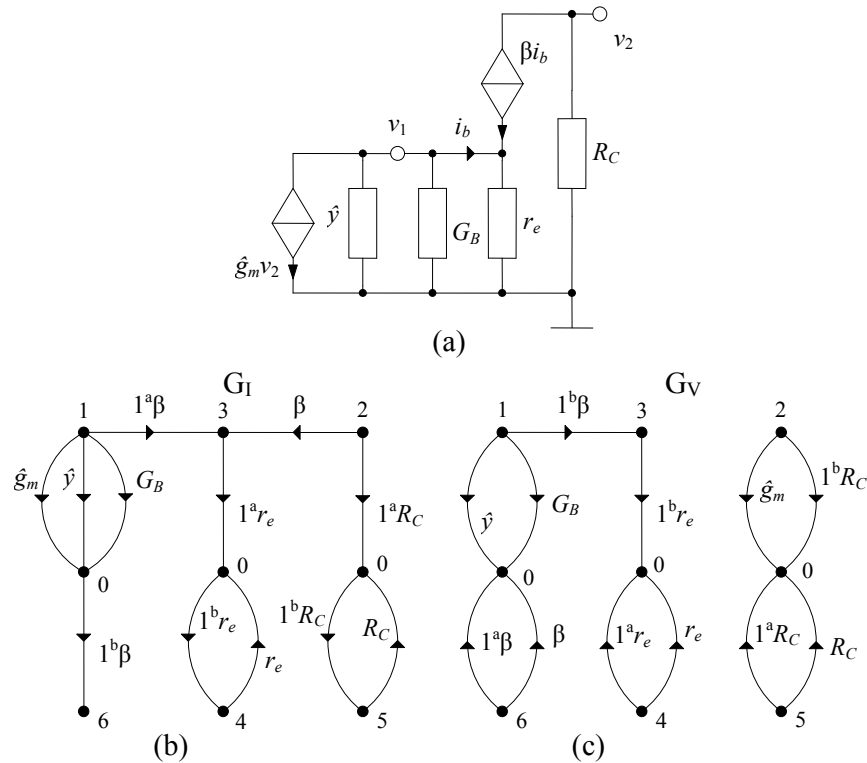


Figure 6: Circuit with resistors and CCCS (a) and its modified current (b) and voltage (c) graphs.

Of course, for such an elementary circuit the required transfer function can be calculated by inspection as $T_v = -\frac{\beta}{\beta + 1} \frac{R_C}{r_e}$. Note that three circuit components: R_C , r_e and the CCCS βi_b , could not be directly handled by the classical two-graph approach.

Using the element stamps from **Table 1**, the current and voltage graphs are constructed, as shown in **Fig. 6(b, c)**, respectively. These graphs have only six common trees, which are listed in **Fig. 7** with appropriate signs:

- , \hat{g}_m , R_C , β , $1^a r_e$, $1^b \beta$, $1^b r_e$
- + , \hat{y} , r_e , β , $1^a R_C$, $1^b \beta$, $1^b R_C$
- + , \hat{y} , r_e , $1^a \beta$, $1^a R_C$, $1^b \beta$, $1^b R_C$
- + , G_B , r_e , β , $1^a R_C$, $1^b \beta$, $1^b R_C$
- + , G_B , r_e , $1^a \beta$, $1^a R_C$, $1^b \beta$, $1^b R_C$
- + , $1^a \beta$, $1^a R_C$, $1^a r_e$, $1^b \beta$, $1^b R_C$, $1^b r_e$

Figure 7: All common spanning trees, with signs, of the two graphs in **Fig. 6**.

After removing terms $1^a x$ and $1^b x$ from the common trees and sorting the product terms with respect to the unique symbols \hat{y} and \hat{g}_m , we can write the determinant of the MNAM of the augmented circuit as

$$\hat{\Delta} = \Delta + \hat{y}\Delta_{11} + \hat{g}_m\Delta_{12} = 1 + G_B r_e + G_B r_e \beta + \hat{y}(r_e + r_e \beta) - \hat{g}_m R_C \beta \quad (10)$$

Now, the required transfer function can be obtained as a ratio of two cofactors:

$$T_v \doteq \frac{v_2}{v_1} = \frac{\Delta_{12}}{\Delta_{11}} = \frac{-R_C \beta}{r_e + r_e \beta} = -\frac{\beta}{\beta + 1} \frac{R_C}{r_e} \quad (11)$$

It can be easily confirmed that other transfer functions, namely:

$$Z_{in} = \frac{\Delta_{11}}{\Delta} = \frac{(\beta + 1)r_e}{1 + (\beta + 1)G_B r_e}, \quad R_T = \frac{\Delta_{12}}{\Delta} = \frac{-\beta R_C}{1 + (\beta + 1)G_B r_e} \quad (12)$$

are also calculated correctly using the proposed modification of the two-graph method.

The major advantages of the two-graph method are: (i) there are no term cancellations, (ii) all types of circuit components can be handled and (iii) efficient techniques exist to generate common spanning trees in decreasing order of magnitude. The last point is of great importance to the approximate symbolic analysis of large analog circuits, where only a relatively small number of common spanning trees with significant contribution to the solution need to be generated. An outline of this application will be presented in the next section.

5. Application of the Two-Graph Method to Symbolic Approximation

All symbolic analysis methods that have a goal of generating the fully s -expanded form of (1) suffer from the well known complexity problem: the exponential, $O(a^n)$, or superexponential, $O(n^n)$, growth of the number of terms with the circuit size. It has been suggested that such exact symbolic analysis of a randomly chosen circuit is infeasible if the sum of the number of nodes and the number of branches is greater than 40 [4].

One solution to the complexity problem is to generate an approximate transfer function with only the most significant terms. The two-graph method is very suitable for this task because there are efficient techniques for generating the spanning trees in decreasing order of their weights [13]. (The weight of a tree is defined as a sum of weights of its branches. Since we want to compare the products of admittances, a natural way of dealing with the problem is to make the weight of each branch equal to the logarithm of the magnitude of its corresponding admittance.)

The process starts with generation of the *maximum tree*, *i.e.*, the tree with the largest weight. A simple algorithm of Kruskal [14] or Prim [15] can be employed. Once the maximum tree is known, each of its branches is exchanged with all suitable links (this is called the *T-exchange*; a valid T-exchange must produce another spanning tree). This generates a list of trees that is guaranteed to contain the next largest tree. The T-exchange process is then repeated for the second largest tree, producing the third largest tree, and so on. The procedure is stopped when the simplified transfer function approximates the exact function to the required accuracy [3]. Of course, the exact transfer function must be calculated numerically beforehand. The above approach can be used only to approximate the transfer function as a whole for a fixed frequency $s = j\omega$.

Example 3

To illustrate the process of computer generation of simplified symbolic expressions, consider a DC small-signal (augmented) equivalent circuit of a feedback amplifier, shown in Fig. 8(a). We want to obtain approximate (to better than 1%) expressions for its voltage transmittance and input resistance. Readers would recognize the circuit to be an inverting amplifier with an imperfect op amp, having finite open-loop gain μ , finite differential input resistance R_d and non-zero output resistance R_o . Using element stamps from Table 1, the current and voltage graphs of the augmented circuit are constructed; they are shown in Fig. 8(b, c). The branch weight is the logarithm of the element's (trans)conductance.

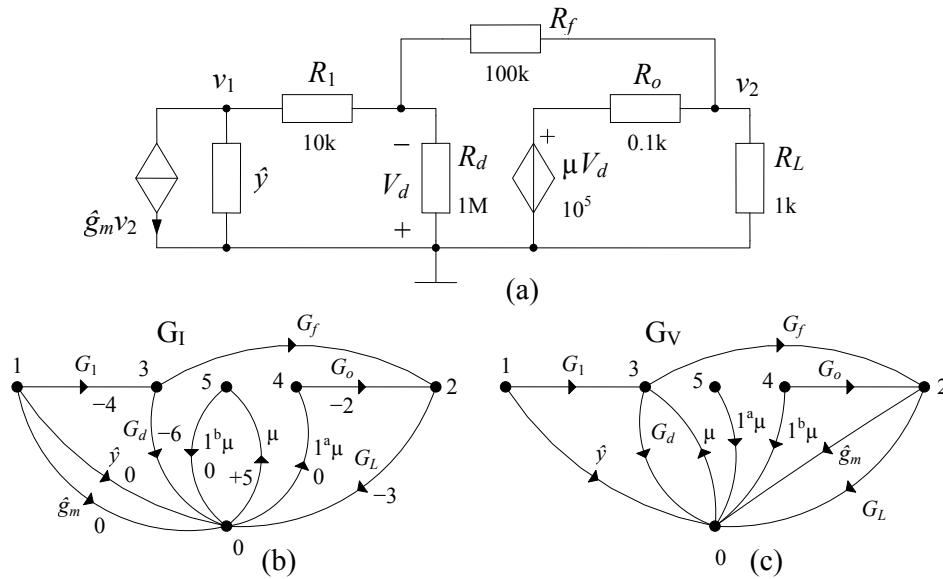


Figure 8: A DC small-signal model of an amplifier with an imperfect op amp (a) and its associated current (b) and voltage (c) graphs.

There are 17 common spanning trees in G_I and G_V (the total number of trees in G_I is 68). The trees, with their weights (converted back to the product of conductances) and signs, are listed in Fig. 9 in decreasing order of weights.

$T_0 \rightarrow 10^{-2}, -, \hat{g}_m, \mu, G_1, G_o, 1^a \mu$	$T_{10} \rightarrow 10^{-9}, +, \hat{y}, G_d, G_L, 1^a \mu, 1^b \mu$
$T_1 \rightarrow 10^{-3}, +, \hat{y}, \mu, G_f, G_o, 1^a \mu$	$T_{11} \rightarrow 10^{-11}, +, \hat{y}, G_d, G_f, 1^a \mu, 1^b \mu$
$T_2 \rightarrow 10^{-6}, +, \hat{y}, G_1, G_o, 1^a \mu, 1^b \mu$	$T_{12} \rightarrow 10^{-11}, +, G_1, G_f, G_o, 1^a \mu, 1^b \mu$
$T_3 \rightarrow 10^{-7}, +, \hat{y}, G_1, G_L, 1^a \mu, 1^b \mu$	$T_{13} \rightarrow 10^{-12}, +, G_1, G_d, G_o, 1^a \mu, 1^b \mu$
$T_4 \rightarrow 10^{-7}, +, \hat{y}, G_f, G_o, 1^a \mu, 1^b \mu$	$T_{14} \rightarrow 10^{-12}, +, G_1, G_f, G_L, 1^a \mu, 1^b \mu$
$T_5 \rightarrow 10^{-7}, +, \mu, G_1, G_f, G_o, 1^a \mu$	$T_{15} \rightarrow 10^{-13}, +, G_1, G_d, G_L, 1^a \mu, 1^b \mu$
$T_6 \rightarrow 10^{-8}, +, \hat{y}, G_d, G_o, 1^a \mu, 1^b \mu$	$T_{16} \rightarrow 10^{-15}, +, G_1, G_d, G_f, 1^a \mu, 1^b \mu$
$T_7 \rightarrow 10^{-8}, +, \hat{y}, G_f, G_L, 1^a \mu, 1^b \mu$	
$T_8 \rightarrow 10^{-9}, +, \hat{g}_m, G_1, G_f, 1^a \mu, 1^b \mu$	
$T_9 \rightarrow 10^{-9}, +, \hat{y}, G_1, G_f, 1^a \mu, 1^b \mu$	

Figure 9: All common spanning trees, with weights and signs, of the two graphs in Fig. 8.

The required symbolic expression for the network functions are formulated by successively adding terms to the numerator and the denominator as they are generated in decreasing order of magnitude. After the first three steps we notice that no more significant contributions to the voltage transmittance expression can be made as the

weight of T_2 is already three orders of magnitude smaller than the weight of T_1 . So, the approximate voltage transmittance formula can now be obtained as

$$T_v = \frac{-\mu G_1 G_o + \cancel{\dots}}{\mu G_f G_o + \cancel{G_1 G_o} + \cancel{\dots}} \approx \frac{-\mu G_1 G_o}{\mu G_f G_o} = -\frac{G_1}{G_f} = -\frac{R_f}{R_1} \quad (13)$$

The approximate expression for the input resistance will be obtained after nine steps, when the weights of generated trees are 100 times smaller than the first term in the denominator of the formula. So, the approximate input resistance of the amplifier is

$$R_{in} = \frac{\mu G_1 G_o + \dots}{\mu G_1 G_f G_o + \dots} \approx \frac{1}{G_1} = R_1 \quad (14)$$

As expected, the approximate expressions are exactly the ones used by every electronics engineer. They have been derived, however, by a completely automatic process.

The two-graph tree enumeration approach can be also used to approximate the coefficients at individual powers of s . To obtain the approximate coefficient at s^k , the trees with exactly k capacitors must be generated in decreasing order of weights [16]. This can be accomplished by the use of *matroid intersection algorithms* [17]. Again, the numerical values of the coefficients must be calculated beforehand. For relatively small circuits a simple numerical interpolation technique can be employed [10]. For large circuits, with many tens of capacitors, an adaptive interpolation technique was developed [18].

6. Two-Graph Method Applied to Parameter Extraction Using Hybrid Equation

In Sections 4 and 5 we have shown how the two-graph method can be applied to symbolic calculation of the determinant of a (modified) node admittance matrix. The two-graph method has also other interesting applications. In this section we will show how it can be used to formulate and solve the hybrid (tableau) equation to obtain semisymbolic expressions for network functions (this process is also called *parameter extraction*) [5]. For simplicity of presentation we will consider only the RLC- g_m circuits. As shown in section 4.2, the method can be extended to other circuit components.

As before, a network will be represented topologically by two linear graphs: the voltage graph G_V and the current graph G_I . We choose a common spanning tree containing only impedance elements (R, sL). All admittance elements (G, g_m, sC) are in the cotree. The hybrid system of equations for such network, with respect to the chosen tree T , is

$$\mathbf{H}\mathbf{x} = \begin{bmatrix} \mathbf{I} & \mathbf{0} & -\mathbf{Z}_T & \mathbf{0} \\ \mathbf{B}_T & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{Q}_C \\ \mathbf{0} & -\mathbf{Y}_C & \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{v}_T \\ \mathbf{v}_C \\ \mathbf{i}_T \\ \mathbf{i}_C \end{bmatrix} = \mathbf{0} \quad (15)$$

The first and the last rows of (15) consist of the v-i relationships for the tree (T) and the cotree (C) elements; the second and third rows consist of the fundamental loop and the fundamental cutset equations for G_V and G_I , respectively. \mathbf{B}_T is the fundamental loop matrix in G_V and \mathbf{Q}_C is the fundamental cutset matrix in G_I . The matrices \mathbf{Z}_T and \mathbf{Y}_C are diagonal; \mathbf{I} is an identity matrix of an appropriate order.

Let the circuit have n nodes and b branches and contain k symbolic components (Y_1^S, \dots, Y_k^S) in the co-tree branches (links) and l symbolic components (Z_1^S, \dots, Z_l^S) in the tree branches; we define $w = b - n - k + 1$, $t = n - l - 1$. Diagonal matrices \mathbf{Y}_C and \mathbf{Z}_T can be partitioned as follows:

$$\mathbf{Y}_C = \begin{bmatrix} \mathbf{Y}_C^S & \mathbf{0} \\ \mathbf{0} & \mathbf{Y}_C^N \end{bmatrix}, \quad \mathbf{Z}_T = \begin{bmatrix} \mathbf{Z}_T^S & \mathbf{0} \\ \mathbf{0} & \mathbf{Z}_T^N \end{bmatrix} \quad (16)$$

where superscript S denotes immitances of symbolic components and superscript N denotes immitances of components given numerically.

Matrices \mathbf{B}_T and \mathbf{Q}_C can also be partitioned as follows:

$$\mathbf{B}_T = \begin{bmatrix} \mathbf{B}_{11} & \mathbf{B}_{12} \\ \mathbf{B}_{21} & \mathbf{B}_{22} \end{bmatrix}, \quad \mathbf{Q}_C = \begin{bmatrix} \mathbf{Q}_{11} & \mathbf{Q}_{12} \\ \mathbf{Q}_{21} & \mathbf{Q}_{22} \end{bmatrix} \quad (17)$$

Rows of \mathbf{B}_{11} and \mathbf{B}_{12} correspond to symbolic cotree branches (in G_V) and their columns correspond to symbolic and numeric tree branches, respectively. Rows of \mathbf{B}_{21} and \mathbf{B}_{22} correspond to numeric cotree branches. Rows of \mathbf{Q}_{11} and \mathbf{Q}_{12} correspond to symbolic tree branches (in G_I) and their columns correspond to symbolic and numeric co-tree branches, respectively. Rows of \mathbf{Q}_{21} and \mathbf{Q}_{22} correspond to numeric tree branches. The submatrices are therefore of the following order: \mathbf{B}_{11} : $k \times l$, \mathbf{B}_{22} : $w \times t$, \mathbf{Q}_{11} : $l \times k$, \mathbf{Q}_{22} : $t \times w$.

Let $S_x = \{1, 2, \dots, x\}$. For a given matrix \mathbf{F} of order $a \times b$ let $\mathbf{F}(I_u, J_v)$ be the submatrix of \mathbf{F} consisting of the rows and columns indexed by the integers in the sets I_u, J_v , respectively. The sets $I_u = \{i_1, i_2, \dots, i_u\}$ and $J_v = \{j_1, j_2, \dots, j_v\}$ are subsets of S_a and S_b , respectively. Let us also introduce the following notation:

$$\mathbf{1}_d^c = \text{diag}[e_1 \ e_2 \ \dots \ e_d]; \quad c < d$$

$$e_x = \begin{cases} 0 & \text{for } x \in \{1, 2, \dots, c\} \\ 1 & \text{for } x \in \{c+1, c+2, \dots, d\} \end{cases} \quad (18)$$

The determinant of the system matrix \mathbf{H} in Eq. (24.18), when some parameters take fixed numerical values, is:

$$\det \mathbf{H} = a + \sum_{J_v} b(\alpha_v) Z_{j_1}^S Z_{j_2}^S \dots Z_{j_v}^S + \sum_{I_u} c(\beta_u) Y_{i_1}^S Y_{i_2}^S \dots Y_{i_u}^S$$

$$+ \sum_{I_u} \sum_{J_v} d(\alpha_v, \beta_u) Z_{j_1}^S Z_{j_2}^S \dots Z_{j_v}^S Y_{i_1}^S Y_{i_2}^S \dots Y_{i_u}^S \quad (19)$$

where the summations are taken over all possible symbol combinations α_v (symbolic tree elements) and β_u (symbolic cotree elements), and the numerical coefficients are given by:

$$\begin{aligned}
a &= \det \left[\mathbf{1}_w + \mathbf{B}_{22}^N (-\mathbf{Q}_{22}^N) \right] = \det \left[\mathbf{1}_t + (-\mathbf{Q}_{22}^N) \mathbf{B}_{22}^N \right] \\
b(\alpha_v) &= \det \left(\mathbf{1}_{t+v}^v + \begin{bmatrix} -\mathbf{Q}_{12}(J_v, I_w) \\ -\mathbf{Q}_{22}^N \end{bmatrix} \begin{bmatrix} \mathbf{B}_{21}^N(I_w, J_v) & \mathbf{B}_{22}^N \end{bmatrix} \right) \\
c(\beta_u) &= \det \left(\mathbf{1}_{w+u}^u + \begin{bmatrix} \mathbf{B}_{12}(I_u, J_t) \\ \mathbf{B}_{22}^N \end{bmatrix} \begin{bmatrix} -\mathbf{Q}_{21}^N(J_t, I_u) & -\mathbf{Q}_{22}^N \end{bmatrix} \right) \\
d(\alpha_v \beta_u) &= \det \left(\mathbf{1}_{v+t+u}^{v+t+u} + \begin{bmatrix} -\mathbf{Q}_{11}(J_v, I_u) & -\mathbf{Q}_{12}(J_v, I_w) \\ -\mathbf{Q}_{21}^N(J_t, I_u) & -\mathbf{Q}_{22}^N \\ \mathbf{1}_u & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{B}_{11}(I_u, J_v) & \mathbf{B}_{12}(I_u, J_t) & -\mathbf{1}_u \\ \mathbf{B}_{21}^N(I_w, J_v) & \mathbf{B}_{22}^N & \mathbf{0} \end{bmatrix} \right)
\end{aligned} \tag{20}$$

In the above equations, $\mathbf{0}$ represents a zero matrix of an appropriate order and the submatrices \mathbf{B}_{ij}^N and \mathbf{Q}_{ij}^N are defined as:

$$\begin{aligned}
\mathbf{B}_{21}^N(I_w, J_v) &= \mathbf{Y}_C^N \mathbf{B}_{21}(I_w, J_v), & \mathbf{B}_{22}^N &= \mathbf{Y}_C^N \mathbf{B}_{22} \\
\mathbf{Q}_{21}^N(J_t, I_u) &= \mathbf{Z}_T^N \mathbf{Q}_{21}(J_t, I_u), & \mathbf{Q}_{22}^N &= \mathbf{Z}_T^N \mathbf{Q}_{22}
\end{aligned} \tag{21}$$

where the submatrix $\mathbf{B}_{21}(I_w, J_v)$ is obtained from the submatrix \mathbf{B}_{21} by including all of its rows and only columns corresponding to a particular combination (α_v) of symbolic tree elements; submatrix $\mathbf{Q}_{21}(J_t, I_u)$ is obtained from the submatrix \mathbf{Q}_{21} by including all its rows and only columns corresponding to a particular combination (β_u) of symbolic co-tree elements.

Application of (19) and (20) for a circuit with m symbolic parameters requires, theoretically, the calculation of 2^m determinants. Not all of these determinants may need to be calculated due to the following property of the determinants in (20). If a set of symbolic tree elements (α_v) forms a cutset in G_1 (*symbolic tree cut-set*), then the corresponding coefficients $b(\alpha_v)$ and $d(\alpha_v \beta_u)$ in (19) equal to zero. Likewise, if the set of symbolic cotree elements (β_u) forms a loop in G_V (*symbolic co-tree loop*), the corresponding coefficients $c(\beta_u)$ and $d(\alpha_v \beta_u)$ in (19) equal to zero.

Once the determinant $\det(\mathbf{H})$ is obtained from (19), the sorting scheme, identical to that described in p.3.2, is applied and the required network function(s) can be calculated using (4) and (6).

The main feature of this approach is the fact that each coefficient at a valid symbol combination is obtained directly by calculating a single, easily formulated determinant (a polynomial in s , in general case). The method was implemented in a computer program UTSSNAP. The following example illustrates this technique of parameter extraction.

Example 4

In the circuit in **Fig. 10(a)** only two components, R_1 and g_m , are given symbolically. We want to find the input impedance Z_{in} in a semi-symbolic form using the parameter extraction method based on the two graph tableau formulation.

Since we want only to calculate the input impedance, the circuit augmentation can be limited to the admittance \hat{y} . The voltage and current graphs of the augmented circuit are shown in **Fig. 10(b)**. The common spanning tree chosen is $T = \{R_1, R_2, R_3\}$ with one symbolic element. For this circuit we have: $n = 4$, $b = 7$, $k = 2$, $l = 1$, $w = 2$ and $t = 2$.

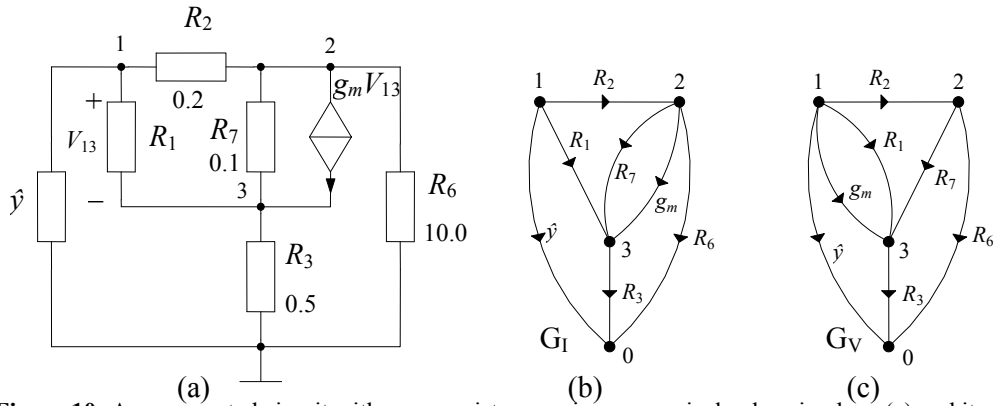


Figure 10: An augmented circuit with some resistances given numerical values in ohms (a) and its current (b) and voltage (c) graphs.

The matrices \mathbf{Y}_C , \mathbf{Z}_T , \mathbf{Q}_C and \mathbf{B}_T , can now be determined as:

$$\mathbf{Y}_C = \left[\begin{array}{c|c} \hat{y} & \\ \hline g_m & \\ \hline & 0.1 \\ & \hline & 10 \end{array} \right] \quad \mathbf{Z}_T = \left[\begin{array}{c|c} R_1 & \\ \hline & 0.2 \\ & \hline & 0.5 \end{array} \right] \quad (22)$$

$$\mathbf{Q}_C = \left[\begin{array}{cc|cc} 1 & 1 & 1 & 1 \\ 0 & -1 & -1 & -1 \\ \hline 1 & 0 & 1 & 0 \end{array} \right] \quad \mathbf{B}_T = \left[\begin{array}{c|cc} -1 & 0 & -1 \\ -1 & 0 & 0 \\ \hline -1 & 1 & -1 \\ -1 & 1 & 0 \end{array} \right]$$

Using (21) we can calculate matrices \mathbf{B}_{22}^N and \mathbf{Q}_{22}^N :

$$\mathbf{B}_{22}^N = \begin{bmatrix} 0.1 & 0 \\ 0 & 10 \end{bmatrix} \begin{bmatrix} 1 & -1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 0.1 & -0.1 \\ 10 & 0 \end{bmatrix}$$

$$\mathbf{Q}_{22}^N = \begin{bmatrix} 0.2 & 0 \\ 0 & 0.5 \end{bmatrix} \begin{bmatrix} -1 & -1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} -0.2 & -0.2 \\ 0.5 & 0 \end{bmatrix} \quad (23)$$

Now, applying (20), the coefficient a in (19) is calculated as:

$$a = \det \left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 0.1 & -0.1 \\ 10 & 0 \end{bmatrix} \begin{bmatrix} 0.2 & 0.2 \\ -0.5 & 0 \end{bmatrix} \right)$$

$$= \det \begin{bmatrix} 1.07 & 0.02 \\ 2 & 3 \end{bmatrix} = 3.17 \quad (24)$$

Since there is only one symbolic tree element, namely R_1 , we have: $\alpha_v = \{R_1\}$ and the associated sets: $J_v = \{1\}$, $I_w = \{1,2\}$. Using (21) we calculate:

$$\mathbf{B}_{21}^N(I_w, J_v) = \mathbf{Y}_{Cr} \mathbf{B}_{21}(I_w, J_v) = \begin{bmatrix} 0.1 & 0 \\ 0 & 10 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \end{bmatrix} = \begin{bmatrix} -0.1 \\ -10 \end{bmatrix} \quad (25)$$

The coefficient $b(R_1)$ can be now obtained from:

$$\begin{aligned}
b(R_1) &= \det \left(\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} -1 & -1 \\ 0.2 & 0.2 \\ -0.5 & 0 \end{bmatrix} \begin{bmatrix} -0.1 & 0.1 & -0.1 \\ -10 & 10 & 0 \end{bmatrix} \right) \\
&= \det \begin{bmatrix} 10.1 & -10.1 & 0.1 \\ -2.02 & 3.02 & -0.02 \\ 0.05 & -0.05 & 1.05 \end{bmatrix} = 10.6
\end{aligned} \tag{26}$$

Other numerical coefficients in (19) are calculated in similar way:

$$c(\hat{y}) = 1.51, c(g_m) = 0, c(\hat{y}g_m) = 0, d(R_1\hat{y}) = 8.12, d(R_1g_m) = 1.05, d(R_1\hat{y}g_m) = 0.51$$

Adding all terms, sorting according to (6) and applying (4) finally results in:

$$Z_{in} = \frac{1.51 + 8.12R_1 + 0.51R_1g_m}{3.17 + 10.6R_1 + 1.05R_1g_m} \tag{27}$$

Matrices in (20) may contain terms dependent on the complex frequency s . Determinants of such matrices are polynomials in s as long as all matrix elements are of the form: $a = \alpha + s\beta$. An interpolation method may be used to calculate the coefficients of those polynomials [10, 18].

7. Synthesis of Electronic Circuits by the Two-Graph Method

7.1. Introduction

The discovery of new active RC circuits has been accomplished by using a variety of techniques. These include proposing a particular circuit topology, analyzing it and matching coefficients (*e.g.* [19, 20]), applying a transformation to a known circuit (*e.g.* [6, 21, 22]), computer generation of all possible topologies [23] and transformation from symbolic transfer function to the active RC circuit by admittance matrix expression [24]. However, in many practical situations engineers prefer to use the prototype LC filter structure as a basis of the active RC filter synthesis. Thus, in this section we present a method of synthesis of the active RC circuits on the basis of the prototype LC networks.

Our method makes prior assumption about circuit topology of the prototype LC filters: they should have the structure of R_T, L_T, C_C circuits (*i.e.*, containing resistors and inductors which can be placed only in tree branches, R_T, L_T , and capacitors which can be placed only in cotree branches, C_C). This restriction allows us to describe the circuit by the hybrid equation (15). The matrix \mathbf{H} in (15) has only ones on the main diagonal, so its determinant will have one term equal to 1. Since all the terms of the determinant must have the same dimension, it follows that all other terms must be dimensionless. This means that they must be the products of equal number of impedances and admittances. It is convenient to think of such products as the products of admittance-impedance pairs. The topological matrices \mathbf{B}_T and \mathbf{Q}_C in (15) determine how the elements are paired, which pairs can appear in the product term, and what is the sign of each term [25].

For such network we calculate the topological matrices \mathbf{B}_T and \mathbf{Q}_C and the symbolic transmittance in s -expanded form. Then we build the equivalent topological matrices without inductive elements. These new matrices form the basis to derive a circuit with resistors, capacitors, nullators and norators which has the same or approximate (for simpler circuits) network function as the prototype LC circuit.

7.2. Transformation of LC filters to active RC circuits

7.2.1. Replacing the topological matrices of the prototype LC filter by the topological matrices of an RC circuit

The networks to be considered are connected, linear, time-invariant LC filters modeled with R_T, L_T, C_C elements. The topological matrices for such networks can be partitioned as shown in **Table 2** (for notational convenience in the remainder of this section we will use matrix $\mathbf{T}_C = -\mathbf{Q}'_C$ ($-\mathbf{Q}_C$ transposed) rather than matrix \mathbf{Q}_C itself).

Table 2: Partitioning of Topological Matrices for the Prototype LC Filters.

$(\mathbf{B}_T)_{LC}$	sL_T	R_T	$(\mathbf{T}_C)_{LC}$	sL_T	R_T
sC_C	\mathbf{B}_{C_C, L_T}	\mathbf{B}_{C_C, R_T}	sC_C	\mathbf{T}_{C_C, L_T}	\mathbf{T}_{C_C, R_T}

The form of individual product pairs, obtained from the topological matrices in **Table 2**, is determined by the symbolic product of their respective row and column labels. The pairs may have one of the two forms: $s^2 C_x L_x$ or $s C_x R_y$. In order for a pair to be a valid symbol combination in a transfer formula, the corresponding entries in the topological matrices must be both equal to ± 1 .

Each quadratic term $s^2 C_x L_y$ may be replicated by in an appropriate configuration of two capacitors and two resistors $C_{x1}, C_{x2}, R_{y1}, R_{y2}$ in an equivalent RC circuit. **Tables 3** and **4** show the structure of relevant topological sub-matrices for the prototype LC circuit and the equivalent RC circuit.

Table 3: Topological Sub-Matrices Related to Quadratic Terms in the Prototype LC Filter.

$(\mathbf{B}_{C_C, L_T})_{LC}$	sL_y	$(\mathbf{T}_{C_C, L_T})_{LC}$	sL_y
sC_x	1	sC_x	1

Table 4: Topological Sub-Matrices Related to Quadratic Terms in the Equivalent RC Circuit.

$(\mathbf{B}_{C_C, R_T})_{RC}$	R_{y1}	R_{y2}	$(\mathbf{T}_{C_C, R_T})_{RC}$	R_{y1}	R_{y2}
sC_{x1}	1	0	sC_{x1}	0	1
sC_{x2}	0	1	sC_{x2}	-1	0

This configuration of capacitors and resistors will create only a quadratic term ($s^2 C_{x1} C_{x2} R_{y1} R_{y2}$) in the transfer function, just like the inductor L_y and the capacitor C_x ($s^2 L_y C_x$). Single terms of the form $s C_x R_y$ are not valid in the topology implied by matrices in **Table 4**, since for every possible combination of $s C_x R_y$ there is one zero element in either $(\mathbf{B}_T)_{RC}$ or $(\mathbf{T}_C)_{RC}$.

There are other combinations of RC elements which give raise to quadratic terms. Altogether we have 16 such possibilities, but only three give different solutions. The two remaining combinations have the forms shown in **Tables 5** and **6**.

Table 5: Second Combination of RC Elements, Generating Quadratic Terms.

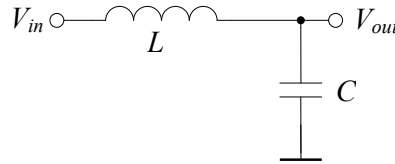
$(\mathbf{B}_{C_C, R_T})_{RC}$	R_{y1}	R_{y2}	$(\mathbf{T}_{C_C, R_T})_{RC}$	R_{y1}	R_{y2}
sC_{x1}	1	0	sC_{x1}	0	-1
sC_{x2}	0	1	sC_{x2}	1	0

Table 6: Third Combination of RC Elements, Generating Quadratic Terms.

$(\mathbf{B}_{C_c, R_T})_{RC}$	R_{y1}	R_{y2}	$(\mathbf{T}_{C_c, R_T})_{RC}$	R_{y1}	R_{y2}
sC_{x1}	-1	0	sC_{x1}	0	1
sC_{x2}	0	1	sC_{x2}	-1	0

Example 5

Consider the following prototype low-pass LC filter:

**Figure 11:** The prototype low-pass LC filter.

The topological matrices $(\mathbf{B}_T)_{LC}$ and $(\mathbf{T}_C)_{LC}$ of this circuit have the following form:

$$\begin{array}{c|c}
 (\mathbf{B}_T)_{LC} & \begin{array}{c} sL \\ \hline sC \end{array} \\
 \hline
 & \begin{array}{c} \boxed{1} \\ \boxed{1} \end{array}
 \end{array}
 \quad
 \begin{array}{c|c}
 (\mathbf{T}_C)_{LC} & \begin{array}{c} sL \\ \hline sC \end{array} \\
 \hline
 & \begin{array}{c} \boxed{1} \\ \boxed{1} \end{array}
 \end{array}$$

The voltage transfer function of the filter in **Fig. 11** is:

$$T_{LC}(s) = \frac{V_{out}}{V_{in}} = \frac{1}{s^2 LC + 1} \quad (28)$$

If we built an equivalent RC circuit with the first combination of resistors and capacitors (**Table 7**), we obtain the following matrices:

$$\begin{array}{c|c}
 (\mathbf{B}_T)_{RC} & \begin{array}{c} R_1 \\ \hline R_2 \end{array} \\
 \hline
 & \begin{array}{c} \boxed{1} \\ \boxed{1} \end{array}
 \end{array}
 \quad
 \begin{array}{c|c}
 (\mathbf{T}_C)_{RC} & \begin{array}{c} R_1 \\ \hline R_2 \end{array} \\
 \hline
 & \begin{array}{c} \boxed{0} \\ \boxed{-1} \end{array}
 \end{array}$$

For these matrices we have an equivalent network function:

$$T_{RC}(s) = \frac{V_{out}}{V_{in}} = \frac{1}{s^2 C_1 C_2 R_1 R_2 + 1} \quad (29)$$

If $C_1 C_2 R_1 R_2 = LC$, then the transfer functions $T_{LC}(s)$ and $T_{RC}(s)$ are identical.

7.2.2. Construction of an equivalent RC circuit with switches

We have shown that the terms of the form $a_2 s^2$ in the transfer function of the prototype LC filters may be replaced by terms of the form $\bar{a}_2 s^2$ in the transfer function of the RC circuit. The question now is how to construct an active RC circuit starting from the topological matrices of the LC filter.

The derivation of the RC circuit with switches consists of two stages starting from the original topological sub-matrices $(\mathbf{B}_{C_c, L_T})_{LC}$ and $(\mathbf{T}_{C_c, L_T})_{LC}$:

1. Each pair of the corresponding non-zero elements of the sub-matrices $(\mathbf{B}_{C_c, L_T})_{LC}$ and $(\mathbf{T}_{C_c, L_T})_{LC}$ (**Table 3**) should be replaced by one of the three possible (2×2) sub-matrices, shown in **Tables 4-6**.

- Each of the two capacitors C_{x1} , C_{x2} and two resistors R_{y1}, R_{y2} are connected to the original circuit with switches. These switches have different positions for the voltage (G_V) and the current (G_I) graphs, such as to realize the topological matrices $(\mathbf{B}_T)_{LC}$ and $(\mathbf{T}_C)_{LC}$ of the prototype LC circuit.

Example 6

Consider again the low-pass LC filter from **Fig. 11**. If we want to obtain an RC circuit with the topological matrices $(\mathbf{B}_T)_{RC}$ and $(\mathbf{T}_C)_{RC}$ given in the Example 5, the circuit should be constructed as shown in **Fig. 12(a)**. Two other circuits, realizing the topological matrices in **Tables 5** and **6**, are shown in **Fig. 12(b)** and **Fig 12(c)**, respectively.

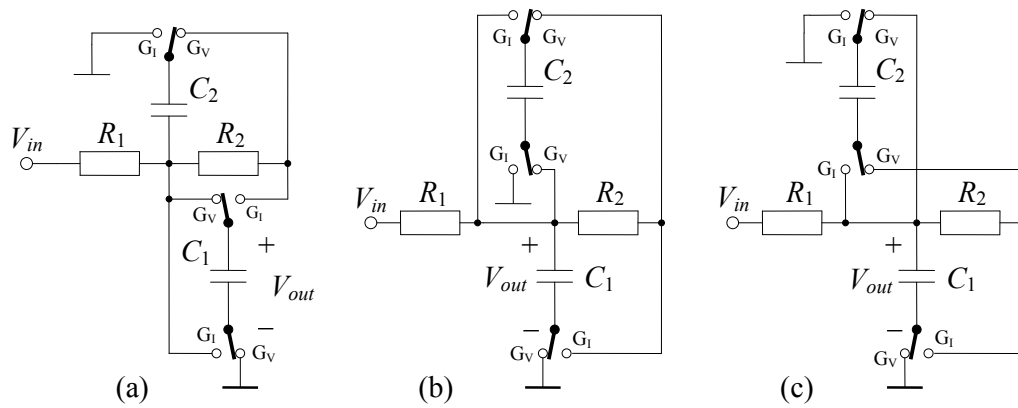


Figure 12: The equivalent circuits to the low-pass LC filter prototype from **Fig. 11**, realizing topological matrices in Table 4 (a), Table 5 (b) and Table 6 (c).

7.2.3. Construction of an equivalent RC circuit with nullors

There are elements which can work as switches in the above circuits. Examining the two-graph stamps for the nullator and norator, given in **Table 1**, we see that they have exactly the required representations in the voltage and current graphs. The nullator can be considered to be a switch which is in the open-circuit position in the current graph and in the short-circuit position in the voltage graph. Similarly, the norator can be considered to be a switch which is in the short-circuit position the current graph and in the open-circuit position in the voltage graph.

For example, if the switches in the circuit in **Fig. 12(a)** are replaced with nullators and norators, we obtain the circuit shown in **Fig. 13(a)**.

A nullator and a norator must appear as a pair which has been given the name *nullor*. Because nullors can be used to represent a variety of different active elements such as BJT, FET, op-amp, current conveyor, voltage follower, current follower, operational transconductance amplifier (OTA), etc. [26], they provide a unified framework for analysis and synthesis of active network. A nullor approximation with both input and output ports floating is called a Four Terminal Floating Nullor (FTFN) [27]. A schematic symbol of the FTFN is shown in **Fig. 13(b)**.

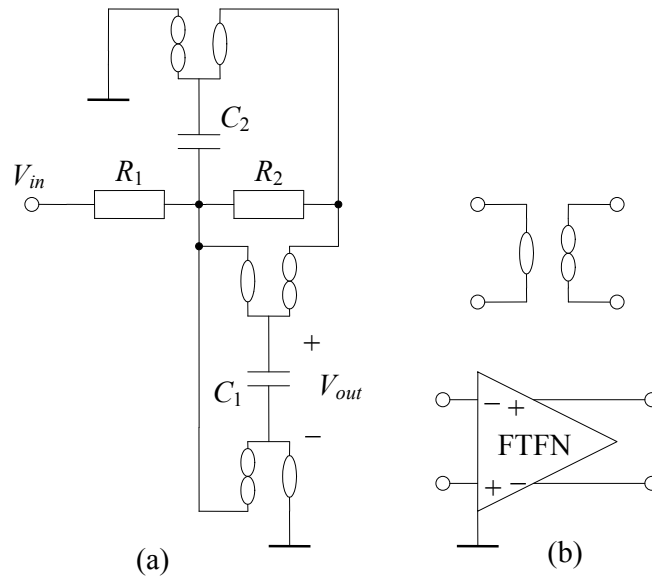


Figure 13: The circuit from **Fig. 12(a)** with the switches replaced by nullators and norators (a). A nullator-norator pair can be approximated by the Four Terminal Floating Nullor (b).

The RC circuit with the FTFNs, equivalent to the prototype LC low-pass filter is shown in **Fig. 14**.

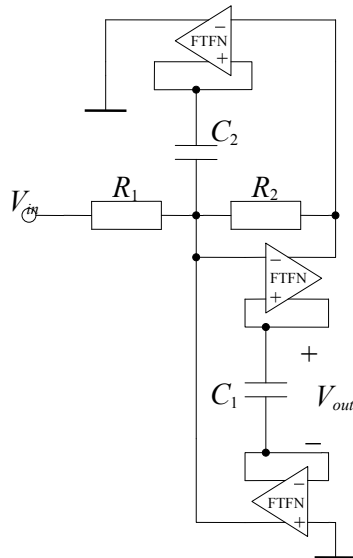


Figure 14: The circuit from **Fig. 13(a)** with the nullator-norator pairs replaced by the FTFNs.

7.2.4. *Simplification of structures of active-RC circuits with nullors*

A method has been described that leads to the topology of an active RC circuit, starting from the prototype LC filter structure. It is clear from the examples that the synthesis process gives a transfer function fully equivalent to that of the prototype LC filter. However, the structure of the active RC circuits may be complicated by the presence of many switches. If we want to obtain simpler topologies, we must accept some additional terms in the transfer function of the equivalent RC circuit. A switch is eliminated if we allow a capacitor to be connected to the same vertices in G_1 and

G_V . This results in an additional ± 1 appearing in the relevant sub-matrices of $(\mathbf{B}_T)_{RC}$ and $(\mathbf{T}_C)_{RC}$. **Table 7** shows a modification that will result in elimination of switches connecting C_{x1} to the circuit..

Table 7: Modified Topological Sub-Matrices of the RC Circuit.

$(\mathbf{B}_{C_C,RT})_{RC}$	R_{y1}	R_{y2}
sC_{x1}	1	1
sC_{x2}	0	1

$(\mathbf{T}_{C_C,RT})_{RC}$	R_{y1}	R_{y2}
sC_{x1}	1	1
sC_{x2}	-1	0

This new configuration of the capacitors and resistors leads to a term of the form as^2+bs ($s^2C_{x1}C_{x2}R_{y1}R_{y2}+sC_{x1}R_{y1}+sC_{x1}R_{y2}$) in the denominator of the transfer function. Thus, the resulting transmittance will only be an approximation of the prototype.

Example 7

Once again, consider the low-pass filter in **Fig. 11**. If we build an equivalent RC circuit with the resistor-capacitor combinations in **Table 7**, its topological matrices will have the form:

$(\mathbf{B}_T)_{RC}$	R_1	R_2
sC_1	1	1
sC_2	0	1

$(\mathbf{T}_C)_{RC}$	R_1	R_2
sC_1	1	1
sC_2	-1	0

From these matrices we obtain the network function of the equivalent circuit:

$$T_{RC}(s) = \frac{V_{out}}{V_{in}} = \frac{1}{s^2 C_1 C_2 R_1 R_2 + s C_1 (R_1 + R_2) + 1} \tag{30}$$

This network function is no longer identical with the prototype as it contains an additional term $sC_1(R_1+R_2)$. However, the circuit is simpler, as shown in **Fig. 15(a)**. A realization of the modified RC filter, with the switch replaced by a FTFN is shown in **Fig. 15(b)**.

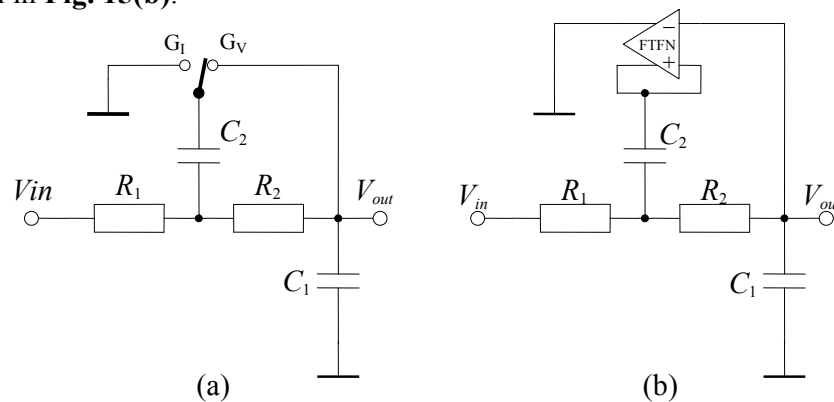


Figure 15: Modified RC filter with a single switch (a) and its realization with the FTFN (b).

The additional term in (30) has a simple circuit interpretation. It represents losses in the inductor, as shown in **Fig. 16**. Expression (31) gives the transfer function of this lossy LC filter. Expressions (30) and (31) are equivalent if $LC = C_1C_2R_1R_2$ and $r = R_1+R_2$.

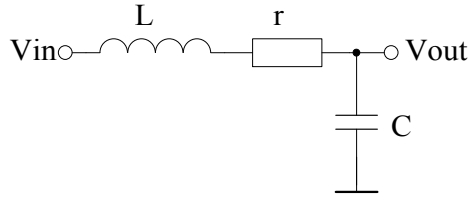


Figure 16: The low-pass LC filter with losses.

$$T_{LC}(s) = \frac{V_{out}}{V_{in}} = \frac{1}{s^2 CL + sCr + 1} \quad (31)$$

If (30) is to be a good approximation of the prototype transfer function (28), then we need $C_1(R_1+R_2) \ll C_1C_2R_1R_2$. This implies that either C_2 or R_1R_2 are very large. Such solutions may not be practical as large capacitances and resistances are expensive (in terms of the ‘real estate’ on the IC chip).

This disadvantage may be removed by using another form of the topological sub-matrices $(\mathbf{B}_{C_c,L_T})_{RC}$ and $(\mathbf{T}_{C_c,L_T})_{RC}$, shown in Table 8. Circuit topology implied by these matrices will lead to a transfer formula containing only a single valid pair: $sC_{x1}R_{y2}$.

Table 8: Alternative Topological Sub-Matrices of the RC Circuit.

$(\mathbf{B}_{C_c,R_T})_{RC}$	R_{y1}	R_{y2}	$(\mathbf{T}_{C_c,R_T})_{RC}$	R_{y1}	R_{y2}
sC_{x1}	1	1	sC_{x1}	0	1
sC_{x2}	0	1	sC_{x2}	-1	0

Example 8

If we build an equivalent RC circuit with the alternative combination of resistors and capacitors (Table 8), we obtain the following topological matrices for the modified RC circuit:

$(\mathbf{B}_T^V)_{RC}$	R_1	R_2	$(\mathbf{T}_C^I)_{RC}$	R_1	R_2
sC_1	1	1	sC_1	0	1
sC_2	0	1	sC_2	-1	0

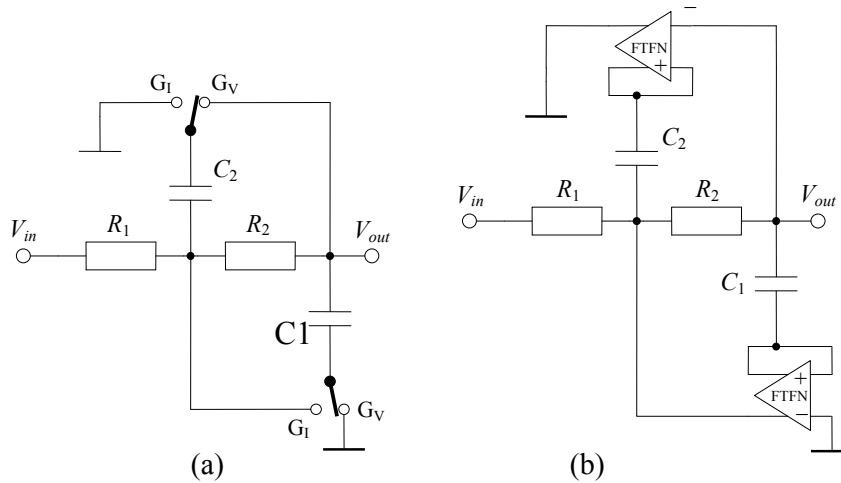


Figure 17: The equivalent RC filter with losses and its realization with the FTFNs.

For these matrices we obtain the following network function, approximating the prototype low-pass LC filter:

$$T_{RC}(s) = \frac{V_{out}}{V_{in}} = \frac{1}{s^2 C_1 C_2 R_1 R_2 + s C_1 R_2 + 1} \quad (32)$$

In this case we have an additional term sC_1R_2 in the network function. However, the circuit configuration is simpler (**Fig. 17**) and we can independently design the equivalent inductor's value L by choosing the product C_2R_1 and the losses by choosing the value of R_2 .

8. Conclusion

The two-graph method has been largely used in symbolic analysis of linear, time-invariant circuits. The method is very suitable for such analysis, since, intrinsically, no cancelling terms are generated. Term generation is based on enumeration of the common spanning trees in the two graphs, describing the circuit topology. Originally, the two-graph method was developed for an important but limited class of RLC- g_m networks. This limitation can be removed by introducing circuit modifications, similar to the ones used in the modified nodal analysis. However, there is a price we have to pay for the ability to analyze a wider class of networks: the modifications usually increase the circuit complexity.

All analysis methods that attempt to generate an exact fully symbolic single formula for a network function suffer from the complexity problem. The number of terms in the expression grows exponentially, $O(a^n)$, or superexponentially $O(n^n)$, with the circuit size. One solution to the complexity problem is to relax the requirement that the formula must be exact and include only those terms that significantly contribute to the final result. The two-graph method is very well suited to this task because algorithms exist that allow the common spanning trees to be generated in decreasing order of magnitude.

Often we are interested in finding the contribution of only a few components to the overall circuit response. If those components are left as symbols and all the others are replaced by their numerical values, the resulting formula will have mixed symbolic and numeric terms. Such expression is called semi-symbolic, and the process of its formulation is known as parameter extraction. The two-graph method is also very effective in parameter extraction. In this chapter we have presented one such application that allows calculation of numerical coefficients at each symbol combination in a single step and with no matrix inversion.

The two-graph method can also be used in network synthesis. We have described an application of the two-graph method to synthesize active RC filters on the basis of the prototype LC filter structures. The use of the Four Terminal Floating Nullor (FTFN) allows an easy transformation from the LC filter structure to the architecture of the RC circuit. The number of available choices is quite manageable. Thus, the method provides a tool for circuit innovation.

References

- [1] W. Mayeda, *Graph theory*. New York: Wiley-Interscience, 1972.
- [2] C. Ho, A.E. Ruehli and P.A. Brennan, "The modified nodal approach to network analysis", *IEEE Transactions on Circuits and Systems*, vol. 25, no. 6, pp. 504-509, 1975.
- [3] F.V. Fernández, P. Wambacq, G. Gielen, A. Rodríguez-Vázquez and W. Sansen, "Symbolic analysis of large analog integrated circuits by approximation during expression generation", in *IEEE International Symposium on Circuits and Systems*, 1994, pp. 25-28.

- [4] E. Henning, "Symbolic approximation and modeling techniques for analysis and design of analog circuits", PhD thesis, University of Kaiserslautern. Aachen: Shaker Verlag, 2000.
- [5] M. Pierzchała and B. Rodanski, "Direct calculation of numerical coefficients in semi-symbolic circuit analysis", in *International Workshop on Symbolic Methods and Applications to Circuit Design*, 1998, pp. 173-176.
- [6] M. Pierzchała and M. Fakhfakh, "Novel structures of RC-active filters for tapped capacitor resonant circuits", in *European Conference on Circuit Theory and Design*, 2009, pp. 129-132.
- [7] P.-M. Lin, *Symbolic Network Analysis*. Amsterdam: Elsevier, 1991.
- [8] G. Gielen and W. Sansen, *Symbolic analysis for automated design of analog integrated circuits*. Boston: Kluwer, 1991.
- [9] H.N. Gabow and E.W. Myers, "Finding all spanning trees of directed and undirected graphs", *SIAM Journal on Computing*, vol. 7, no. 3, pp. 280-287, 1978.
- [10] J. Vlach and K. Singhal, *Computer methods for circuit analysis and design*. NEW York: Van Nostrand Reinhold, 1994.
- [11] B. Rodanski, "Extension of the two-Graph method for symbolic analysis of circuits with non-admittance elements", in *International Workshop on Symbolic Methods and Applications to Circuit Design*, 2002, pp. 17-20.
- [12] H.J. Carlin, "Singular network elements", *IEEE Transactions on Circuit Theory*, vol. 11, pp. 67-72, 1964.
- [13] H.N. Gabow, "Two algorithms for generating weighted spanning trees in order", *SIAM Journal on Computing*, vol. 6, no. 1, pp. 139-150, 1977.
- [14] J.B. Kruskal, "On the shortest spanning subtree of a graph", *Proceedings of the American Mathematical Society*, no. 1, pp. 48-50, 1956.
- [15] R.C. Prim, "Shortest connection networks and some generalizations", *Bell Systems Technical Journal*, no. 36, pp. 1389-1401, 1957.
- [16] Q. Yu and C. Sechen, "Efficient approximation of symbolic network functions using matroid intersection algorithms", *IEEE Transactions CAD of Integrated Circuits and Systems*, vol. 16, no. 10, pp. 1073-1081, 1997.
- [17] H.N. Gabow and R.E. Tarjan, "Efficient algorithms for a family of matroid intersection problems", *Journal of Algorithms*, vol. 5, pp. 8-131, 1984.
- [18] F.V. Fernández, O. Guerra, D.J. Rodríguez-García and A. Rodríguez-Vazquez, "Symbolic analysis of large analog integrated circuits: The numerical reference generation problem", *IEEE Transactions on Circuits and Systems II*, vol. 45, no. 10, pp. 1351-1361, 1998.
- [19] W. Tangsirat, T. Dumawipata, S. Unhavanich and W. Surakamponorn, "Simulation of electronically tunable lossless floating inductor using current-controlled differential current voltage conveyors", in *International Symposium on Communications and Information Technologies*, 2004, pp. 39-42.
- [20] B. Maundy, S. Gift and P. Aronhime, "A novel hybrid active inductor", *IEEE Transactions on Circuits and Systems II, Express Briefs*, vol. 54, no. 8, pp. 663-667, 2007.
- [21] A. Corlosena and G.S. Moschyts, "Nullators and norators in voltage to current mode transformation", *International Journal on Circuit Theory and Applications*, vol. 21, no. 4, pp. 421-424, 1993.
- [22] D.G. Haig, "Some network transformations by terminal interchange", in *IEEE International Symposium on Circuits and Systems*, 1978, pp. 416-421.
- [23] R. Cabeza and A. Carlosena, "On the use of symbolic analyzers in circuit synthesis", *Analog Integrated Circuits and Signal Processing*, vol. 25, no. 1, pp. 67-75, 2000.
- [24] D.G. Haigh, "A method of transformation from symbolic transfer function to active-RC circuit by admittance matrix expansion", *IEEE Transactions on Circuits and Systems I, Regular Papers*, vol. 53, no. 12, pp. 2715-2728, 2006.
- [25] M. Pierzchała and B. Rodanski, "Compact form of s -expanded symbolic network functions for linear electronic circuits", in *International Workshop on Symbolic and Numerical Methods, Modeling and Applications to Circuit Design*, 2008, pp. 12-16.
- [26] E. Tlelo-Cuautle, C. Sánchez-López and F. Sandoval-Ibarra, "Computing symbolic expressions in analog circuits using nullors", *Computación y Sistemas*, vol. 9, no. 2, pp. 119-132, 2005.
- [27] H. Schmid, "Approximating the universal active element", *IEEE Transactions on Circuit and Systems II*, vol. 47, no. 11, pp. 1160-1169, 2000.

Approximation Techniques in Symbolic Circuit Analysis

Francisco V. Fernández^{1,*}, Carlos Sánchez-López², Rafael Castro-López³ and Elisenda Roca-Moreno³

¹Dept. Electronics and Electromagnetism, University of Sevilla and IMSE-CNM, CSIC; ²IMSE-CNM, CSIC and University of Sevilla, Spain, and ³IMSE-CNM, CSIC and University of Sevilla

Abstract: Symbolic circuit analysis suffers from the exponential growth of expression complexity with circuit size. Therefore, either if the symbolic expressions are used for gaining insight into circuit operation or for repetitive computer-based evaluations, simplification becomes mandatory. This chapter reviews the different existing techniques for symbolic expression simplification, classifying them into three categories according to the step at which the simplification is performed: on the circuit equations, during the solution of the circuit equations or after the circuit equations have been solved. Pros and cons of each approach are discussed.

Keywords: Approximated symbolic analysis, approximation techniques, simplification before generation, graph reduction, matrix reduction, simplification after generation, simplification during generation, approximation error, symbolic term generation, two-graph method, matroid intersection, determinant decision diagram.

1. Introduction

Symbolic analysis refers to circuit analysis in which all or part of the circuit parameters are kept as symbols. Symbolic analysis tools have been typically restricted to linear models, therefore limiting their application to linear(ized) or weakly non-linear analysis.

In this chapter we will concentrate on small-signal frequency domain analysis. We can distinguish between fully symbolic analysis and semi-symbolic analysis. In the former case, all component parameters and the complex frequency variable are maintained as symbols. In the latter, some or all component parameters are given numerical values and the rest are maintained as symbols. Our discussion will focus on fully symbolic analysis, since, from the perspective of this chapter, semi-symbolic analysis is considered a particular case of the former.

The analysis of an amplifier like that in **Fig. 1** provides a symbolic expression of the voltage gain with 21 terms:

$$H(s, x) = \frac{g_{m1}g_{m2}r_{\pi1}r_{\pi2}R_3R_L(R_1+R_2)+R_1R_L(R_3+r_{\pi2})g_{m1}r_{\pi1}+R_1R_L(R_3+r_{\pi2})}{g_{m1}g_{m2}r_{\pi1}r_{\pi2}R_3R_LR_1+g_{m1}r_{\pi1}(R_2+R_L)(R_3+r_{\pi2})R_1+(R_2+R_L)(R_3+r_{\pi2})(R_1+r_{\pi1})+(R_3+r_{\pi2})r_{\pi1}R_1} \quad (1)$$

Although this expression is relatively simple, the operation as a feedback amplifier can be better understood if we consider that the first term in numerator and denominator is much larger than the other ones, and if they are neglected, the approximated voltage gain becomes an expression so much easier to interpret:

*Address correspondence to Francisco V. Fernández: Dept. Electronics and Electromagnetism, University of Sevilla and IMSE-CNM, CSIC; E-mail: pacov@imse-cnm.csic.es

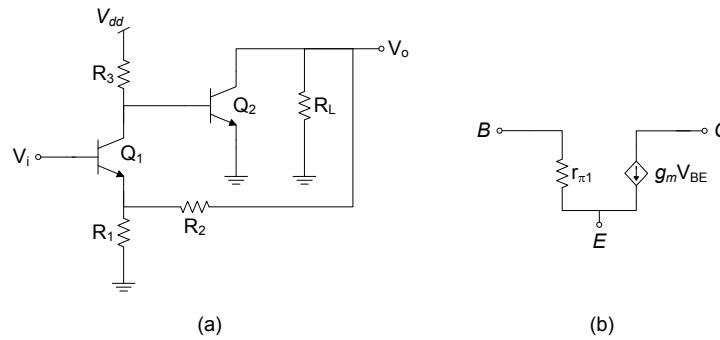


Figure 1: (a) Feedback amplifier; (b) BJT small-signal model.

$$H(s, \mathbf{x}) = \frac{R_1 + R_2}{R_1} \quad (2)$$

Moreover, in case that this expression is to be used in some kind of computer application that requires repetitive evaluations, this can be done much more efficiently.

However, fully symbolic circuit analysis suffers from the exponential growth of expression complexity with circuit size. As Section 4 will show, the number of terms in the symbolic expression grows to about 10^4 terms for a Miller two-stage amplifier with 7 transistors and to more than 10^{10} terms for an opamp with about 20 transistors. This expression complexity not only hampers its interpretation or repetitive evaluation but it can even make their generation computationally impossible. Therefore, the introduction of simplification techniques in symbolic circuit analysis becomes mandatory.

Considering the step of the analysis process at which the simplification is performed we will distinguish three types of techniques:

1. *Simplification Before Generation (SBG) techniques.* The simplification is performed at the circuit model, graph or matrix level, directly on the graph or matrix representing the circuit equations. The goal is to obtain a simplified graph or matrix that can be solved much more efficiently and that yields much simpler symbolic results with a controlled error.
2. *Simplification During Generation (SDG) techniques.* The simplification is applied during the solution process of the circuit equations. The goal is to produce only the most significant part of the symbolic expression, without wasting time in generating symbolic terms with little influence on the final results.
3. *Simplification After Generation (SAG) techniques.* The simplification is performed directly on the symbolic solution. Therefore, it requires the previous circuit analysis and solution of the circuit equations.

Following this classification of techniques, the three remaining sections in this chapter are devoted to them.

2. Simplification After Generation Techniques

As stated above, simplification after generation techniques are applied on the symbolic network functions, once the circuit has been symbolically analyzed. In case these techniques are combined with other simplification (SBG and/or SDG) techniques, they would be the last ones to be applied, but they were the first ones that chronologically appeared, and for this reason, they will be discussed first.

SAG techniques are typically applied on network functions in expanded format:

$$H(s, \mathbf{x}) = \frac{\sum_{j=1}^m s^j f_j(\mathbf{x})}{\sum_{i=1}^n s^i g_i(\mathbf{x})} \quad (3)$$

where the coefficients of the different powers of the complex frequency are sums-of-products of symbolic circuit parameters:

$$\mathbf{x}^T = \{x_1, x_2, \dots, x_Q\} \quad (4)$$

Reported simplification criteria consider simplification at the full frequency range, *i.e.*, the simplification is performed for each coefficient of the complex frequency in numerator and denominator of (3). Let us denote:

$$h_k(\mathbf{x}) = \sum_{l=1}^T h_{kl}(\mathbf{x}) \quad (5)$$

as any coefficient $f_i(\mathbf{x})$ or $g_j(\mathbf{x})$ in (3). The simplification is usually performed by heuristically pruning the insignificant terms in each coefficient $h_k(\mathbf{x})$ of the complex frequency variable in (3) so that an approximate polynomial, $h_{kA}(\mathbf{x})$, is found for each coefficient. This approximate polynomial fits the original one within a user-specified maximum error parameter ε_M inside a given region R of the symbolic parameter space:

$$\max_{\mathbf{x} \in R} \left| \frac{h_k(\mathbf{x}) - h_{kA}(\mathbf{x})}{h_k(\mathbf{x})} \right| < \varepsilon_M \quad (6)$$

Most of the reported approaches perform this fitting only at a single point of the parameter space \mathbf{x}_o , commonly called the *nominal* or *design* point.

The simplification criterion in [1] looks for the largest magnitude term for each coefficient $h_k(\mathbf{x}_o)$ and multiplies it by a user-defined maximum error ε_o , that defines a discrimination threshold. Then, all the terms are taken one by one and those whose magnitude is below the calculated threshold are eliminated. In other words, any term is eliminated from (5) if it fulfills the following condition:

$$|h_{kl}(\mathbf{x}_o)| < \varepsilon_o \cdot \max(|h_{k1}(\mathbf{x}_o)|, |h_{k2}(\mathbf{x}_o)|, \dots, |h_{kT}(\mathbf{x}_o)|) \quad (7)$$

Its main drawback is the lack of control on the accumulated error for each coefficient—the accumulated value of the deleted terms can represent either a small or large part of the total magnitude of each coefficient. Consequently, coefficient errors will probably differ considerably for different coefficients, and large magnitude and phase errors and large pole/zero displacements can be thus expected.

More elaborated criteria require the previous sorting of terms in $h_k(\mathbf{x})$ according to their magnitude at the nominal point \mathbf{x}_o . One possibility is to eliminate the P smallest magnitude terms, P being the largest integer for which the accumulated error is below ε_M [2-4]:

$$\frac{\left| \sum_{l=1}^P h_{kl}(\mathbf{x}_o) \right|}{\left| \sum_{l=1}^T h_{kl}(\mathbf{x}_o) \right|} < \varepsilon_M \quad (8)$$

Mutually canceling terms do not contribute to (8) because they are added with their respective signs. However, such terms may become significant when the simplified formula is evaluated at points other than \mathbf{x}_o . Hence, although this criterion gives very accurate results at \mathbf{x}_o , the resulting error at other points may be well beyond ε_M . This happens, for instance, when mismatches among nominally matched devices are taken into account. Such mismatches have a strong influence on characteristics that rely largely on cancellations, such as the common-mode rejection ratio and the power-supply rejection ratio. In these cases, large insight is gained if explicit mismatch parameters are introduced.

One solution to avoid elimination of mutually canceling terms is to modify (8) as follows:

$$\frac{\sum_{l=1}^P |h_{kl}(\mathbf{x}_o)|}{\sum_{l=1}^T |h_{kl}(\mathbf{x}_o)|} < \varepsilon_M \quad (9)$$

Neglected terms that are of the same order of magnitude as the last one remaining in the simplified expression are recovered and kept in this expression [5].

In the previous criteria, if the same error ε_M were exactly obtained for all numerator and denominator coefficients in (3), the simplified expression would become

$$H(s, \mathbf{x}) = \frac{(1 - \varepsilon_M)f_0(\mathbf{x}) + s(1 - \varepsilon_M)f_1(\mathbf{x}) + \dots + s^m(1 - \varepsilon_M)f_m(\mathbf{x})}{(1 - \varepsilon_M)g_0(\mathbf{x}) + s(1 - \varepsilon_M)g_1(\mathbf{x}) + \dots + s^n(1 - \varepsilon_M)g_n(\mathbf{x})} \quad (10)$$

where there is no change in magnitude or phase and neither zero nor pole displacement. However, expression simplification is a discrete process and, hence, the actual errors are different for each coefficient:

$$H(s, \mathbf{x}) = \frac{(1 - \varepsilon_{M0n})f_0(\mathbf{x}) + s(1 - \varepsilon_{M1n})f_1(\mathbf{x}) + \dots + s^m(1 - \varepsilon_{Mmn})f_m(\mathbf{x})}{(1 - \varepsilon_{M0d})g_0(\mathbf{x}) + s(1 - \varepsilon_{M1d})g_1(\mathbf{x}) + \dots + s^n(1 - \varepsilon_{Mnd})g_n(\mathbf{x})} \quad (11)$$

This may lead to significant root displacements in circuits where the roots are very sensitive to coefficient variations.

Different solutions have been proposed to overcome these problems. A trivial strategy adds a numerical fitting factor to each simplified coefficient such that the numerical evaluation of the pruned coefficients at the nominal point is made equal to the original ones [4]. This approach has been implemented in different symbolic analyzers and guarantees accuracy at the nominal point, but it does not imply any improvement for points other than nominal.

Another possibility is to monitor the magnitude of each term within each $h_k(\mathbf{x})$ in (3), to avoid eliminating those whose magnitude is greater than the denominator in (9) [2]. Another approach is to use an adaptive ε_M scheme: term-pruning is performed step by step and the pole/zero displacements are monitored at each step so that simplifications can be stopped when such displacements are beyond a user-specified safety margin [3]. Unfortunately, even though this guarantees a low error at the nominal point, it does not ensure good results for different points of the parameter space.

All these criteria have assumed that the frequency remains a symbol. If the system function has to be approximated for a single value of the frequency, f_o , then it must be evaluated for $s = j2\pi f_o$. The problem then reduces to approximating the real and imaginary parts of numerator and denominator and, hence, conceptually it is no different from the approximation of individual coefficients of the network function.

Expression approximation for a bound frequency range using a nominal value approach is not easy. One possibility is to perform the approximation for different frequency points within the given range and include in the final expression every term that is present at least in the simplified expression at one frequency point. Full accuracy, however, is not guaranteed with this approach. Increasing the number of sample frequency points diminishes the likelihood of errors but also diminishes the speed of the algorithm. Sections 3 and 4 will discuss a possible solution to this problem. Although the techniques discussed there can also be directly applied, they will not be discussed here as they have been reported for SBG and SDG approaches.

All previous algorithms perform the approximation at a nominal point and there is no guarantee that the accuracy is high enough at other design points. Parameter variations in practical circuits are usually restricted to bounded regions of the parameter space. One possibility to extend the validity range of the simplified expressions is the repetitive application of any previous criteria to each point (a sufficiently fine grid should be defined) inside the bounded region. However, because dimensions of the parameter spaces of practical circuits are usually very large, this approach is computationally intractable in practice.

A solution was reported in [6]. It is based on the use of ranges of variation [7], *i.e.* it assumes that each symbol (device model variable, product of variables, or sum of products) may take any value inside a given range of variation:

$$y_i \in [y_{iL}, y_{iH}] \quad (12)$$

where y_{iL} and y_{iH} are real numbers and $y_{iL} \leq y_{iH}$.

In this approach, the P least significant terms of each coefficient $h_k(\mathbf{x})$ are eliminated as long as the following condition is satisfied:

$$\frac{U([A_{cL}, A_{cH}])}{L([S_L, S_H])} < \varepsilon_M \quad (13)$$

where $[S_L, S_H]$ represents the range of the sum of all terms in coefficient $h_k(\mathbf{x})$, $[A_{cL}, A_{cH}]$ represents the range of the sum of all terms to be pruned and, $U(\cdot)$ and $L(\cdot)$ are the upper and lower range operators, that return the upper and lower limit of the included range, respectively.

To apply this definition, operators among ranges have to be defined:

- **Product of ranges.** Given the multiplication of two symbolic factors, y_i and y_j , the range of their product is:

$$y_i y_j \in [\min(y_{iL} y_{jL}, y_{iL} y_{jH}, y_{iH} y_{jL}, y_{iH} y_{jH}), \max(y_{iL} y_{jL}, y_{iL} y_{jH}, y_{iH} y_{jL}, y_{iH} y_{jH})] \quad (14)$$

- **Addition of ranges.** For a given sum of two symbols, y_i and y_j , the range of the sum is computed by adding the corresponding bounds of the addends:

$$y_i + y_j \in [y_{iL} + y_{jL}, y_{iH} + y_{jH}] \quad (15)$$

- **Modulus of ranges.** For a given symbolic parameter, product of symbols, or sum of products for which a range $[y_{iL}, y_{iH}]$ is defined or calculated, the modulus of ranges operator yields another range, defined from the previous one by taking the modulus of the extremes in an appropriate order:

$$[|y_{iL}|, |y_{iH}|] = [\min(|y_{iL}|, |y_{iH}|), \max(|y_{iL}|, |y_{iH}|)] \quad (16)$$

- **Reciprocal of a range.** For a given symbol y_i whose range does not include zero, its reciprocal is defined as

$$\frac{1}{y_i} \in \left[\frac{1}{y_{iH}}, \frac{1}{y_{iL}} \right] \quad (17)$$

The major drawback of the variation range technique is that excessively conservative results can be obtained, due to two reasons:

- Some circuit parameters are correlated with each other. If each one is assigned a range, range operators ignore the correlations, and this yields overestimation of the real range.
- Direct substitution of the symbolic parameters with their ranges and the real arithmetic operators with their corresponding interval arithmetic operators yield the so-called natural interval extension of the symbolic polynomials. Its main drawback is that the width of the range may be considerably larger than the real one, because each range bounds may be calculated with the upper range bound of one symbolic parameter at a polynomial term and the lower bound at another. This problem is partially avoided by using other interval extensions, like the mean value interval extension of a function [7] $f(\mathbf{x})$:

$$F_{MV}(\mathbf{X}) = f(\mathbf{m}) + \sum_{i=1}^n D_i F(\mathbf{X})(X_i - m_i) \quad (18)$$

where capital letters denote interval extension, the set $\mathbf{m} = \{m_1, m_2, \dots, m_n\}$ is the vector of mean values of the variables x_i , and $D_i F(\cdot)$ is the interval extension of the function derivative with respect to the i -th variable. This interval extension can be computed using the natural interval extension of the derivatives, or recursively calculated using (18).

The approximation based on ranges of variation becomes especially interesting in case of matching devices. Matching devices are those elements designed to have identical nominal values, *e.g.* differential pairs and current mirrors. Due to process variations mismatches occur between such elements. In some performance characteristics device mismatch plays a dominant role and, therefore, introducing explicit mismatch parameters becomes extremely convenient, *e.g.*, given two matched transistors, M_1 and M_2 , their transconductance is symbolically represented as:

$$g_{m1} \rightarrow g_m + \Delta g_m \quad g_{m2} \rightarrow g_m - \Delta g_m \quad (19)$$

In nominal value approaches [2, 3] mismatch parameters, *e.g.* Δg_m , are handled like any other symbolic parameters, *i.e.*, a numerical value equal to the maximum mismatch value is assigned to such mismatch parameter for approximation purposes. However, this does not correspond to the basic philosophy of a mismatch parameter, as there is no mismatch value that can be assimilated to a nominal value, and even the sign is unknown. But the concept of a mismatch parameter, *i.e.*, a parameter that can take any value between a minimum and maximum values fits perfectly with the range of variation philosophy.

3. Simplification Before Generation Techniques

In this kind of techniques, approximations are performed directly on the network equations, either in the form of a matrix, a graph, or the small-signal equivalent

circuit itself, which are all mainly determined by the solution technique for the network equations that will be applied afterwards.

The complexity reduction of the network equations has a tremendous impact on the complexity of the symbolic results. Therefore, the degree of interpretability introduced is significantly improved. Besides, the computation time and memory requirements of the subsequent SDG/SAG algorithms are considerably reduced due to the exponential relationship between circuit complexity and the number of terms to be generated.

The basic goals of a simplification before generation approach are:

- The simplified system of equations must model the circuit behavior correctly within the error constraints in the specified frequency range.
- The system of equations resulting after the simplification step should be the simplest possible.

To this end two elements are needed:

- An ordering mechanism for the contribution of the different graph branches/nodes or matrix entries appearing in the graph/matrix that represents the system of equations.
- A stopping criterion to decide the number of devices/nodes or matrix entries that can be affected by the simplification without exceeding the error specifications within the defined frequency range.

According to the form of circuit model or equations in which SBG is applied, the next three subsections will deal with matrix-based, graph-based and circuit-based approaches respectively. In Section 3.4 pros and cons of these approaches are discussed and their error control mechanisms are introduced in Section 3.5.

3.1. Matrix-based approaches

A simplification before generation technique at the matrix level was introduced in [8-10]. This technique builds the circuit equations through the indefinite nodal admittance matrix:

$$\mathbf{I} = \mathbf{Y} \cdot \mathbf{V} \quad (20)$$

The network function can be expressed as the ratio of two cofactors (a first-order one and a second-order one) of this matrix. As a first step, device parameters are eliminated from each cofactor of the nodal admittance matrix if the error induced in the cofactor is below a given error threshold. The error induced in one matrix determinant by a modification $\Delta m_{i,j}$ of the value at the location (i, j) is:

$$\Delta(|\mathbf{M}|) = \Delta m_{i,j} \cdot (-1)^{i+j} |\mathbf{M}_{ij}| \quad (21)$$

Matrix \mathbf{M} in (21) is the appropriate cofactor of the nodal admittance matrix, according to the numerator or denominator of the network function, and \mathbf{M}_{ij} is the minor, obtained by deleting the i -th row and the j -th column of \mathbf{M} .

As each circuit element is mapped into four positions on the nodal admittance matrix, parameter elimination can be performed at one, two, or all four positions.

Concurrently with the device parameter elimination, this technique tries to reduce determinant dimension by factoring out rows and columns with only one nonzero entry and performs row and column operations to reduce the number of symbols or nonzero entries. These heuristics do not alter the value of the determinant and are not approximations in fact. However, they are valuable as they partially palliate the cancellation problem of determinant-based approaches.

Capacitors must also be included in the nodal admittance matrix for high-frequency analysis. Then, applied approximations are only valid in a limited frequency range. Three scanning frequencies are considered per decade, used to evaluate the influence of elimination of the corresponding capacitor parameter. One element is eliminated at four, two, or one location, only if the induced error in the determinant is smaller than an error bound for all scanning frequencies.

This approach separately controls the error in the determinants that correspond to the numerator and the denominator of the network function but not in the network function itself. If we consider a generic matrix formulation:

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{b} \quad (22)$$

a simple solution would be to calculate the effect on the elements of interest in vector \mathbf{x} when any matrix entry is removed. This involves a numerical matrix inversion for each removal that is tried.

A nice solution to avoid such repetitive calculation of inverse matrices is to apply the Sherman-Morrison formula that allows calculating the numerical influence in the system output when a matrix entry $a_{i,j}$ is modified by a value $\Delta a_{i,j}$ [11]:

$$a_{i,j}^* = a_{i,j} + \Delta a_{i,j} \quad (23)$$

The perturbation on the k -th element of vector \mathbf{x} can be calculated as [11]:

$$\Delta x_k = \frac{-\Delta a_{ij} \cdot a_{ki}^{(-1)}}{1 + \Delta a_{ij} \cdot a_{ji}^{(-1)}} \mathbf{A}_{j,\cdot}^{-1} \cdot \mathbf{b} \quad (24)$$

where $a_{ji}^{(-1)}$ is the matrix entry at position (j,i) of matrix \mathbf{A}^{-1} and $\mathbf{A}_{j,\cdot}^{-1}$ is the j -th row of \mathbf{A}^{-1} .

According to these perturbation values, a ranking list is generated where each symbol of the matrix is ordered depending on their numerical influence on a requested component (network variable) [11]. Then, starting with the least significant parameter, eliminations are performed while the error keeps below a given threshold. A heuristic to decide if the influence list has to be recalculated is used to avoid a direct re-calculation after each symbol elimination (which would mean a large waste of computational time). It also controls that the matrix that will result after the term elimination is not singular.

3.2. Graph-based approaches

The simplification before generation methodology in [12] is performed using the two-graph approach and has been implemented in two steps. In the first step, the voltage graph and the current graph are built for both the numerator and the denominator of the network function. Each device contribution (represented in the form of an admittance y) to the numerator, $c(y, N)$, and denominator, $c(y, D)$, are calculated, as well as each device complementary contribution, $\bar{c}(y, N)$ and $\bar{c}(y, D)$:

$$\begin{aligned} c(y, N) &= \frac{yN(y;)}{N} & \bar{c}(y, N) &= \frac{N(; y)}{N} \\ c(y, D) &= \frac{yD(y;)}{D} & \bar{c}(y, D) &= \frac{D(; y)}{D} \end{aligned} \quad (25)$$

where $N(y;)$ and $D(y;)$ correspond to the sum of product terms in N and D that contain the admittance y , and $N(; y)$ and $D(; y)$ to the sum of product terms in N and D , that do not contain y .

This step is performed for a set of frequency samples. Devices with small values of $c(y, N)$ or $c(y, D)$ for all frequency samples are weakly contributing devices and are candidates for deletion in the corresponding graphs. Devices with large values of $c(y, N)$ or $c(y, D)$ for all frequency samples are strongly contributing devices and are candidates for contraction of its terminal nodes in the corresponding graphs. When any deletion or contraction is performed, all the remaining contributions are recalculated, and if they change significantly with respect to the original value, the deletion/contraction operation of that element is cancelled.

Then, the graphs are further simplified simultaneously. The dual contraction error, $e_c(y; H)$, and the dual deletion error, $e_d(y; H)$, of $H(s)$ with respect to the device y are defined as [12]:

$$e_c(y; H) = \frac{\frac{N(y;)}{D(y;)} - \frac{N}{D}}{\frac{N}{D}} \quad e_d(y; H) = \frac{\frac{N(; y)}{D(; y)} - \frac{N}{D}}{\frac{N}{D}} \quad (26)$$

Those elements with low error value can be deleted/contracted from both the numerator and the denominator without affecting the network function value too much. Again, before definitively performing any operation (deletion or contraction) the remaining dual errors are analysed to avoid any operation that causes a large variation on other contributions.

Due to the separate simplification performed on the numerator and the denominator in the first step, it may happen that some devices are eliminated in the numerator (denominator) and still appear at the denominator (numerator). As a result of this, it

will be impossible to build a simplified circuit out of the results. Also, the simplification procedure is performed on a set of discrete frequency samples; this means that the accuracy is only guaranteed at those frequency values.

A technique based on a signal flow graph approach was introduced in [13, 14]. This flow graph is composed of voltage nodes (that represent the voltage of the nodes in the circuit) and current nodes (that represent the voltage of the nodes in the circuit multiplied by the sum of all the self-admittances attached to them). Then, the simplification is performed on this network graph by the application of the following set of operations, called network model transformations [13, 14]:

- *Removal of signal path*: deletion of an incoming meta-edge (set of all parallel branches between two nodes) of a summing vertex.
- *Open-loop root removal*: deletion of a meta-edge not belonging to a signal path.
- *Vertex-pair contraction*: short the voltage vertices and current vertices associated to two nodes.
- *Subgraph substitution*: replacement of a part of the graph by other different graph without altering the connectivity of the network graph.

An important point of the analysis of a linear network, N , is related to the calculation of the network graph complexity:

$$C(N) = n_{fol} + n_{sum} + n_{fwp} + n_{fbl} \quad (27)$$

where n_{fol} is the number of open-loop roots, n_{sum} is the number of summing points, n_{fwp} is the number of forward paths and n_{fbl} is the number of feedback loops in the graph.

For each of the four operations described above, taking into account the performance parameters, $p_i(N, X)$, a ranking function is defined:

$$R_p(N, T_j, X) = \frac{[E_p(N, T_j, X)]^\alpha}{[Q_p(N, T_j)]^{1-\alpha}} \quad (28)$$

being X the design point, T_j the j -th network model transformation, and

$$Q_p(N, T_j) = C(N) - C(T_j(N)) \quad (29)$$

the transformation quality factor, that measures the difference between the complexity of the original network model to the one after one simplification step j , α is a tuning factor and,

$$E_p(N, T_j, X) = |P(N', X) - P(N, X)| \quad (30)$$

is the model performance error, that is the weighted norm of the performance deviation vector. This model performance error has to be measured in a set of frequency points that cover the frequency range defined by the user. According to the ranking value defined in (28), the simplification operations are performed until the specified errors are fulfilled.

Like the previous algorithm, the simplification is performed on a finite number of frequency samples. No SDG algorithm has been defined for this methodology, and it is not clear that a methodology of this kind is feasible for this type of graphs.

3.3. Circuit-based approaches

The approach in [15] performs the approximation on the network under analysis directly at the circuit level. It replaces those elements whose contribution (appropriately measured) to the network function is small, by a zero-admittance (element removal) or zero-impedance element (contraction of terminal nodes).

A first question is to decide if node contractions must be prioritized over branch removals or vice versa. After the SBG process, the resulting simplified graph must be solved, commonly by the application of an SDG process. The most efficient SDG algorithms reported are based on the matroid theory (see Section 4) and their computational complexity grows much faster with the number of circuit nodes than with the number of graph branches. Therefore, node contractions are prioritized in the proposed algorithm. The following steps summarize the algorithm operation:

- a) Compute the contribution to the network function of the contraction of the terminal nodes of each device individually and build a sorted list.
- b) Pick the least significant contraction from the list and compute the error.
- c) If the maximum error has not been exceeded, perform the node contraction, remove all devices connected between that pair of nodes, reorder the contraction list and go to step (b); otherwise continue with the next step.
- d) Compute the contribution to the network function of the removal of each branch individually and build a sorted list.
- e) Pick the least significant branch removal from the list and compute the error.
- f) If the maximum error has not been exceeded, perform the branch removal, reorder the removal list and go to step (d); otherwise end the algorithm.

However, the experience with a large number of circuits shows that the order in which nodes are contracted or branches are removed keeps basically the same as the order in which they appear in the contraction/removal sorted list built at steps (a) and (c). As a consequence of this, the time required to perform the simplification can be highly decreased by eliminating the reordering of such lists in steps (c) and (f), without losing accuracy control.

3.4. Comparison of SBG approaches

As stated above, matrix-based approaches are able to perform elimination of matrix entries at one, two or four positions. Elimination at four positions is equivalent to branch deletion in graph-based and circuit-based approaches. Elimination at one or

two positions may provide some extra simplification. However, it decreases the insight on the simplified results since the simplified matrix does not correspond to a circuit with less devices and/or nodes.

The branch contraction operation in graph-based and circuit-based approaches does not have a correspondence on matrix-based approaches. Therefore the complexity of the results after the SBG step may differ between the different techniques from circuit to circuit.

A very important drawback of matrix-based approaches is that the SBG step must usually be followed by a SDG step. The most efficient SDG algorithms reported are based on the two-graph approach (see Section 4), which is obviously not applicable to the results of a matrix-based SBG techniques but it can be applied to the other two categories of techniques.

The circuit-based techniques may yield slightly more complex results as no separate deletion/contraction operations in numerator and denominator are performed. However, it exhibits two advantages: first, no graph or matrix for the circuit has to be built and second, during the simplification process, a direct correspondence with the nodes and devices of the original circuit is kept.

3.5. Error control

In all SBG approaches above, an error control mechanism exists that decides when no more matrix entries can be eliminated, no more nodes can be contracted, or no more branches can be deleted.

The approaches in [8-14] perform the evaluation of the contributions to the network function of the elimination of matrix entries or the successive node contractions and branch removals at a set of frequency samples within the range being considered. However this solution may yield simplified matrices/graphs/circuits that do not fulfil the error specification at some frequency points outside this initial selected set. Besides, a dense sampling, at least in the neighbourhood of the poles and zeros of the system will be necessary because of the significant variations of the network function around these points. Therefore, there are two drawbacks of error control methods based on frequency sampling:

- 1) The correctness (error specs are met in the complete frequency range) of the resulting simplified circuits cannot be guaranteed.
- 2) A dense spectrum of sampled points decreases the possibility of error excess but increases noticeably the computational cost of the algorithm.

As an illustrative example let us consider the integrator in **Fig. 2**. Simplification before generation is applied to this circuit with a magnitude and phase error specification of $\Delta|H| \leq \pm 5dB$ and $\Delta\phi_H \leq \pm 5^\circ$ in the frequency range $1Hz \leq f \leq 100MHz$. The magnitude and phase errors are evaluated at a small set of frequency samples. **Fig. 3** shows the resulting magnitude and phase errors. Dashed lines represent the magnitude and phase error specifications. The solid triangles represent some of the frequency samples. As can be seen, error specifications are met at the frequency samples but are considerably exceeded at other frequency values.

Increasing the number of frequency samples can palliate the problem, but the appropriate number of frequency samples is not known a priori and the computation time grows linearly with the number of samples.

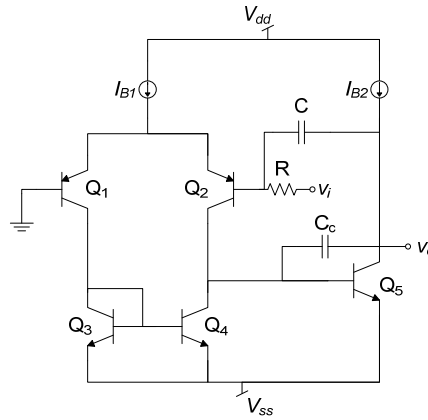


Figure 2: Miller integrator.

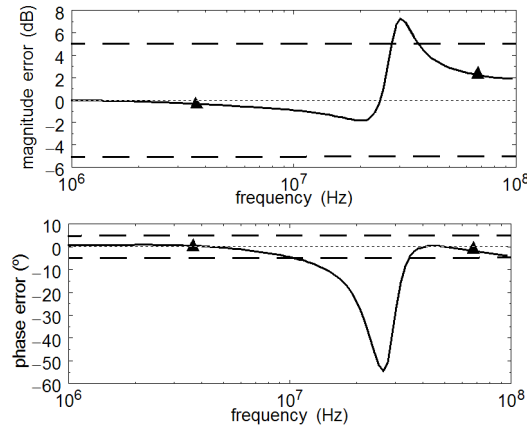


Figure 3: Magnitude and phase errors.

To solve the problems of sampling-based approaches, an interval analysis approach is proposed in [16, 17]. Let us denote $H_{ex}(s)$ the network function of the complete circuit with only the complex frequency s as symbolic parameter, and $H_{ap}(s)$ the analogous network function of a simplified circuit in which the appropriate node contraction(s) and/or device removal(s) have been performed during the application of the SBG algorithm. The magnitude and phase errors are given by:

$$\Delta|H| = \frac{|H_{ex}(j\omega)| - |H_{ap}(j\omega)|}{|H_{ex}(j\omega)|} = 1 - \frac{\sqrt{\frac{N_{apr}^2 + N_{api}^2}{D_{apr}^2 + D_{api}^2}}}{\sqrt{\frac{N_{exr}^2 + N_{exi}^2}{D_{exr}^2 + D_{exi}^2}}} \quad (31)$$

$$\Delta\phi_H = \angle H_{ex}(j\omega) - \angle H_{ap}(j\omega) = \arctan \frac{N_{exi}}{N_{exr}} - \arctan \frac{D_{exi}}{D_{exr}} - \arctan \frac{N_{api}}{N_{apr}} + \arctan \frac{D_{api}}{D_{apr}}$$

where

$$\begin{aligned} H_{ap}(j\omega) &= \frac{N_{ap}(j\omega)}{D_{ap}(j\omega)} = \frac{N_{apr}(j\omega) + jN_{api}(j\omega)}{D_{apr}(j\omega) + jD_{api}(j\omega)} \\ H_{ex}(j\omega) &= \frac{N_{ex}(j\omega)}{D_{ex}(j\omega)} = \frac{N_{exr}(j\omega) + jN_{exi}(j\omega)}{D_{exr}(j\omega) + jD_{exi}(j\omega)} \end{aligned} \quad (32)$$

Therefore, the evaluation of the maximum magnitude and phase errors requires:

- A technique to obtain the network functions $H_{ex}(s)$ and $H_{ap}(s)$ of (usually large) analog circuits with only the complex frequency as symbolic variable.
- An efficient technique to obtain the maxima of the functions in (31) when ω varies within a given range.

The first requirement can be solved very efficiently by using the numerical interpolation technique [18]. One major problem in polynomial interpolation applied to analog integrated circuits is the dramatic effect of round-off errors, due to the finite precision arithmetics of computers. An efficient adaptive scaling solution to solve this problem is proposed in [19].

A trivial solution to the second requirement is to use a set of frequency samples but the same drawbacks exposed in Section 3.5.1 will also appear here. A possible solution is the application of the Newton-Raphson method to find the zeros of the derivatives of (31). However, since the frequency ranges required for symbolic analysis vary over a wide range the search can be slow, some zeros may be skipped or convergence may fail.

A possible solution to this issue is to resort to the interval analysis techniques that were introduced in Section 2. $\Delta|H|$ and $\Delta\phi_H$ in (31) are functions in ω that can take any value within the frequency interval $[\omega_L, \omega_U]$. The problem can be solved if accurate estimates of the lower and upper bounds of $\Delta|H|$ and $\Delta\phi_H$ when the frequency varies in this range can be calculated. As stated in Section 2, the natural interval extension usually overestimates the range of this function [7]. To avoid this, the natural interval extension is not applied to (31) but to the derivatives of (31). Although, the estimates of the derivatives are also very conservative, the zero inclusion in the resulting interval extension is enough to delimit frequency sub-ranges in which the maximum magnitude and phase errors occur. Then, the exact frequency points for which the maximum magnitude or phase errors occur in those frequency sub-ranges are easily calculated by means of the Newton-Raphson method.

This error evaluation technique has been applied to the SBG techniques in Section 3.3. Initially a very small set of frequency samples is selected. For this set of values, the simplification process described in Section 3.3 is performed. Then, the error evaluation technique in this Section is applied to find any possible violation of error constraints within the full frequency range of interest. If the errors are fulfilled, the procedure stops. Otherwise, the frequency point where the magnitude/phase error exhibits its maximum value is added to the initial set of frequency points and one

additional SBG step is performed on it. Each node contraction and branch removal is performed only if the error specifications are met at all frequency points of the set. This process is iteratively applied until the required accuracy is ensured for the complete frequency range defined by the user.

4. Simplification During Generation Techniques

Simplification During Generation (SDG) techniques start from some formulation of the network equations of a circuit in the form of a graph or matrix (*e.g.* voltage and current graphs, directed graphs, *etc.*) and solve them by directly generating the most simplified expression that fulfils some error criteria. Simplification During Generation approaches have two main advantages: first, the analysis is faster, as no time is wasted generating terms that would be neglected if a SAG technique with analogous error specifications were applied on the exact symbolic results. Second, its smaller memory consumption enables the analysis of much larger circuits without exhausting the computer resources.

Typically, SDG algorithms generate symbolic terms in decreasing order of magnitude until the number of terms is enough to model the behavior of the circuit with a given accuracy. Two tasks are involved in this problem: (a) a term generator, *i.e.*, an algorithm that can generate symbolic terms in decreasing order of magnitude, according to their influence on the circuit behavior; and (b) an error control mechanism, that can determine when the generated symbolic expression fulfils the error specifications.

4.1. Term generation

4.1.1. Two-graph approach

The first reliable algorithms capable of efficiently generating terms in decreasing order of magnitude for large circuits were reported in [20-23]. All these algorithms are based on the two-graph method. This method uses two graphs: a voltage graph and a current graph. Given the small-signal model of a circuit, they can be easily built. Each circuit node corresponds to an equivalent node in the voltage and current graphs. Each passive device corresponds to one branch between the graphs nodes that correspond to its terminal nodes. For voltage-controlled current sources, the voltage graph is constructed with the controlled branch and the current graph with the controlling branch. **Fig. 4** shows a sample circuit and the corresponding voltage and current graphs. All other types of controlled sources are converted to the interconnection of passive elements and voltage-controlled current sources, as illustrated in **Fig. 5** [24]. For convenience, inductors are usually replaced by the interconnection of capacitors and voltage-controlled current sources. Each branch is assigned the admittance (or transadmittance) of the corresponding circuit element.

The computation of the most significant terms for each power of the frequency s^k , in the numerator and denominator of the network function, can be expressed as the following graph problem: “Given the voltage and the current graphs of a circuit with n nodes and b branches, enumerate subsets of branches in decreasing order of magnitude that:

- (1) constitute a spanning tree in the voltage graph VG;

- (2) constitute a spanning tree in the current graph CG;
- (3) contain k capacitance branches and $(n - k - 1)$ (trans)conductance branches.”

A valid symbolic term is given by the product of all branch admittances included in each common spanning tree of the voltage and current graphs. The sign of the term is determined separately, using the topological information of both graphs [25]. Although the sign of each term is difficult to obtain by hand-made calculations, it can be very efficiently calculated with a negligible computational time.

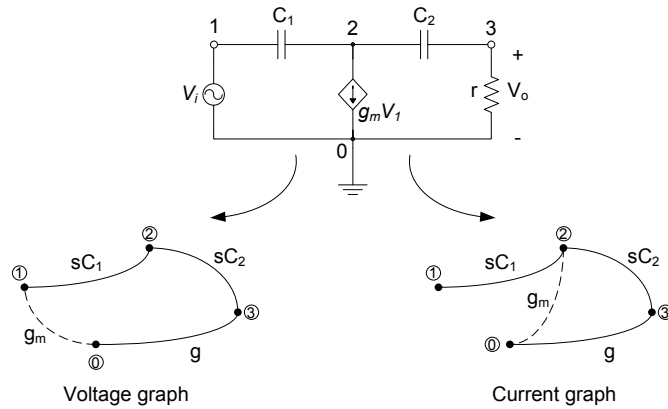


Figure 4: A simple circuit and its corresponding voltage and current graphs.

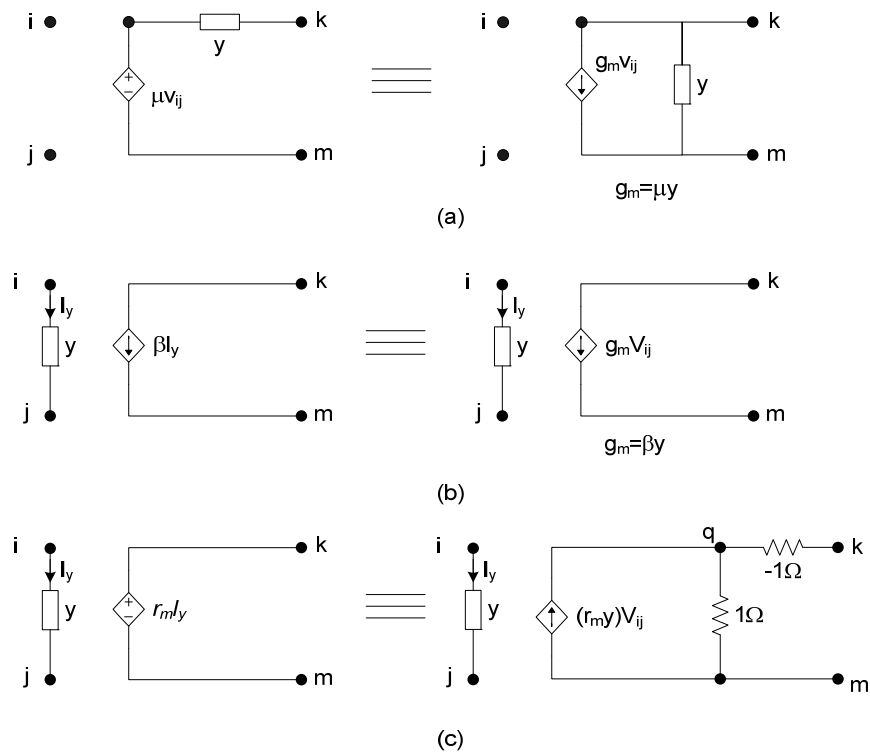


Figure 5: Equivalent circuits for (a) voltage-controlled voltage sources (VCVS), (b) current-controlled current sources (CCCS) and (c) current-controlled voltage sources (CCVS).

The procedure behind the methodologies in [12, 20-23, 26, 27] is the following: first, select one of the graphs, generate spanning trees with k capacitance branches in decreasing order of magnitude in that graph; then, for each tree, check if it is also a spanning tree in the current graph. The generation of spanning trees in decreasing order follows the algorithm in [28], originally proposed for graphs with a single type of branches but extensible to graphs with two types of branches [26].

4.1.2. *Matroid intersection approach*

The term generation problem can be also formulated in terms of the matroid theory [29]. A *matroid* $M = (E, \mathfrak{I})$ is a structure in which E is a finite set of elements and \mathfrak{I} is a family of subsets of E , which satisfy some axioms [29]. A subset I in \mathfrak{I} is an *independent set* of the matroid $M = (E, \mathfrak{I})$. A maximal independent set is a *base* of the matroid. A matroid $M = (E, \mathfrak{I})$ is the *graphic matroid* of graph G if E is the set of branches of G and a subset $I \subseteq E$ is in \mathfrak{I} if and only if I is a cycle-free subset of branches. In a connected graph, a base of its graphic matroid corresponds to a spanning tree.

Let π be a partition that separates E into m disjoint blocks B_1, \dots, B_m , and let d_i ($i=1, \dots, m$) be m non-negative integers. Then, $M = (E, \mathfrak{I})$ is a *partition matroid* if \mathfrak{I} is the family of subsets I that satisfy $|I \cap B_i| \leq d_i$ ($i=1, \dots, m$).

The term generation implies the ranked (in decreasing order of weight) enumeration of bases that are common to three matroids [29], commonly known as a weighted three-matroid intersection problem. For instance, the three matroids involved in the formulation of the tree enumeration problem in the two-graph method described above are: two graphic matroids associated to the voltage and current graphs where each spanning tree is a base of the matroid, and a partition matroid where each set of $n-1$ branches with k capacitances is a base of the matroid.

The formulation of the term generation problem in terms of matroid theory was first proposed in [30], only for the two-graph method. Later on, a formulation and comparison, among the two-graph method, the directed-tree enumeration method, the Coates flow graph method, the Laplace determinant expansion and the parameter extraction method of Sannuti and Puri was introduced in [31, 32].

The intersection of three matroids is considered to be, in general, a NP-hard (as hard as any non-deterministic polynomial time) problem. If a problem can be formulated as a weighted 2-matroid intersection problem, then a polynomial time algorithm exists, being that of Camerini and Hamacher [33] one of the most efficient ones. Moreover, the efficiency can be further improved by exploiting specific properties of the matroids under consideration.

The analysis in [31, 32] is extremely interesting to select the optimum formulation method when the matroid intersection theory is applied. This is illustrated in **Table 1** for the examples depicted in **Fig. 6** [31, 32]. Four different circuits have been analyzed using the different analysis methods. The second and third rows show the number of nodes and devices (after replacement of semiconductor devices by small-signal models) in those circuits. This gives a clear idea of the problem complexity. The fourth row shows the total number of spanning trees in the voltage graph for each

case. The fifth row shows the number of spanning trees in the current graph (normalized to the values in row three). Then, the sixth row shows a lower bound on the number of common spanning trees to both graphs. Finally, the remaining rows show (also normalized to the results in row three) the calculation of the number of terms generated with the different approaches.

Table 1: Number of Terms with Different Analysis Techniques for the Circuits of Fig. 6.

	Fig. 6(a)	Fig. 6(b)	Fig. 6(c)	Fig. 6(d)
# nodes	4	5	9	16
# circuit elements	8	16	31	70
# spanning trees in G_V	36	346	$3.55 \cdot 10^5$	$2.96 \cdot 10^{11}$
# spanning trees in G_I	1	2.13	1.01	2.20
lower bound on # common spanning trees	0.667	0.694	0.553	0.222
exact # different terms	0.722	0.694	0.553	
# terms with directed-tree enumeration	2.08	1.43	1.18	11.9
# terms with signal flow graph	1.92	4.17		
# terms with Coates flow graph / Laplace determinant expansion	2.85	1.20	6.23	5.197
# terms with parameter extraction of Sannuti & Puri	1.22	8.62	195	$2.55 \cdot 10^7$

Assuming that an algorithm with the same efficiency is available to enumerate bases in the matroid intersection problem corresponding to each of the methods in **Table 1**, it can be concluded that the number of bases in the two-graph method is smaller than in the other methods. This means that many bases are generated in other methods that do not correspond to a valid symbolic term in the final symbolic expressions. This ratio tends to increase when the circuit complexity increases. It is also interesting that the number of bases in the voltage graph is smaller than in the current graph.

Moreover, not only the number of bases is smaller, but algorithms that exploit the specific circumstances involved in the problem and achieve a better efficiency than the standard algorithm in [33] have been reported for the two-graph approach.

Therefore, the two-graph method is the best formulation method for an SDG implementation. As the enumeration of common spanning trees for a given coefficient of the network function is a 3-matroid intersection problem and the matroid intersection algorithm in [33] is a 2-matroid intersection algorithm, three possibilities arise:

- a) *Intersect the voltage graphic matroid and the partition matroid and for each enumerated base check if it is also a base in the current graphic matroid.* By exploiting specific features of the two matroids involved, the computational complexity of this procedure can be considerably lowered. This approach has been implemented in [27, 30].

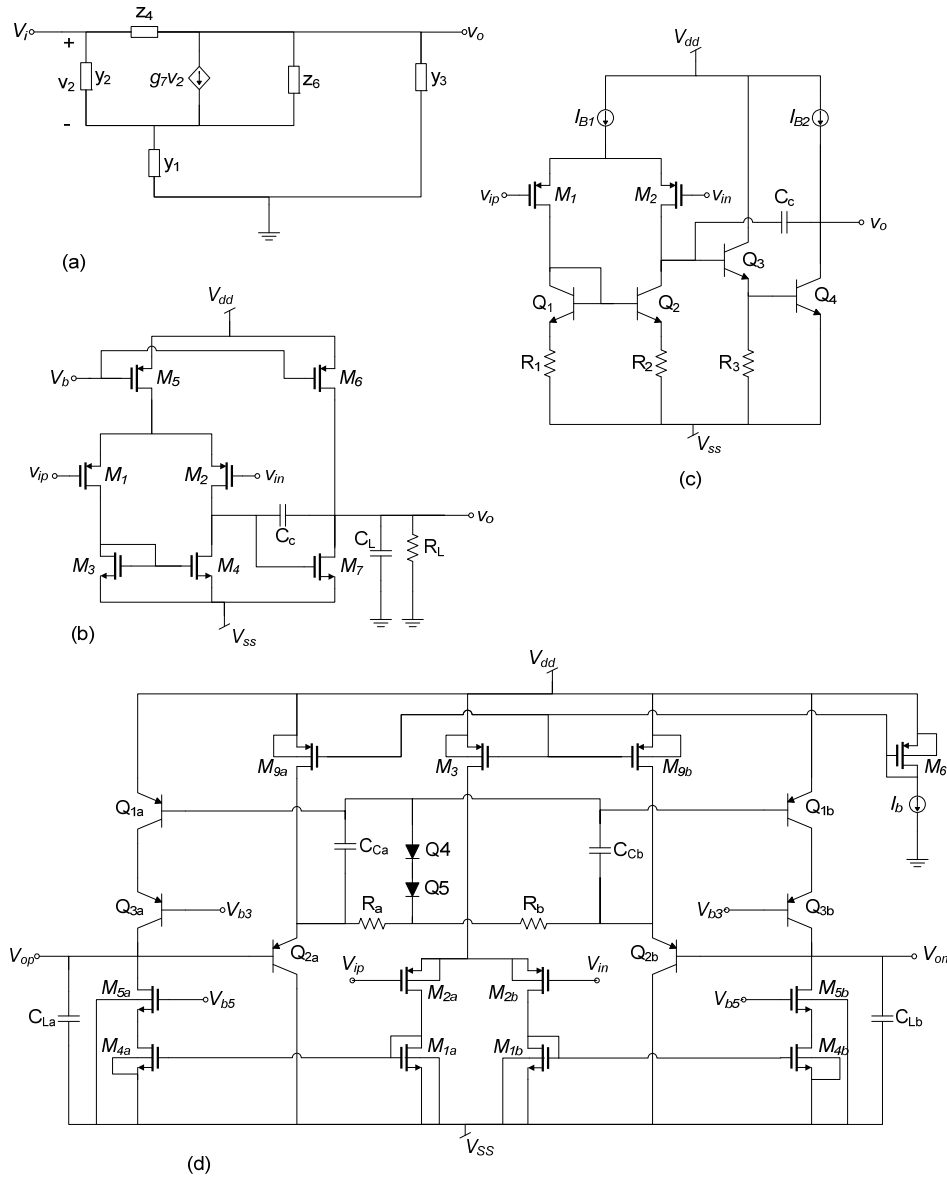


Figure 6: (a) Simple network, (b) CMOS Miller OTA, (c) BiCMOS Miller OTA and (d) BiCMOS fully differential OTA.

This approach has in its efficiency for large circuits its main drawback. This is due to the fact that only a part of the spanning trees generated (common to VG and the partition matroid) will be spanning trees of the current graph. What is more, as the size of the circuit increases, the number of non-valid terms grows fast. Therefore, when the generation process is started, a lot of resources are wasted in the generation of terms that will not be valid spanning trees of the CG.

- b) *Intersect the current graphic matroid and the partition matroid and for each enumerated base check if it is also a base in the voltage graphic matroid.* This is the dual approach to the previous one. However, as demonstrated with the examples in **Table 1**, the number of bases is larger in the current graph, and, therefore, this alternative is disadvantageous with respect to the previous one and will not be paid any more attention.

- c) *Intersect the two graphic matroids and, for each enumerated base, check if it contains the desired number of capacitances.* Common spanning trees of the voltage and current graphs in decreasing order of tree admittance product are generated using for instance the algorithm in [33]. The computational complexity is larger than in the first approach above but, in this case, no time is wasted in generating spanning trees of one of these graphs which, afterwards, are not spanning trees in the other graph. However, no control is provided on the branch type (frequency-dependent or non-frequency dependent elements), and, hence, the weight of capacitor admittances must be evaluated at a fixed frequency.

The advantage of this method is that all the generated terms are valid. However, there exists an important drawback: many terms can be generated more than once (when they are generated at several frequency samples) and this means a loss of efficiency. A definitive answer on the advantages with respect to the first approach above (intersection of voltage graphic matroid and partition matroid) depends on the number of frequency samples, but this is related to the error control methodology that is discussed in section 4.2.

4.1.3. Determinant decision diagram (DDD)-based approach

In this section, we describe another approach that is based on the Determinant Decision Diagram (DDD) concept [34, 35]. DDDs are compact graphs that have been used to represent matrix determinants and they are constructed for the Laplace expansion method. Each coefficient of the admittance matrix is considered as one distinct symbol and each of them is represented into DDDs as a non-terminal vertex along with its sign and labeled as a_i (a to g in Fig. 7) [34]. Further, a DDD has two terminal vertices, namely: *0-terminal* vertex and *1-terminal* vertex, and one starting vertex, called *root* vertex. Each non-terminal vertex has two edges: *1-edge* (solid arrow) and *0-edge* (dashed arrow), and a path from the root vertex to the 1-terminal is called *1-path*, as depicted in Fig. 7.

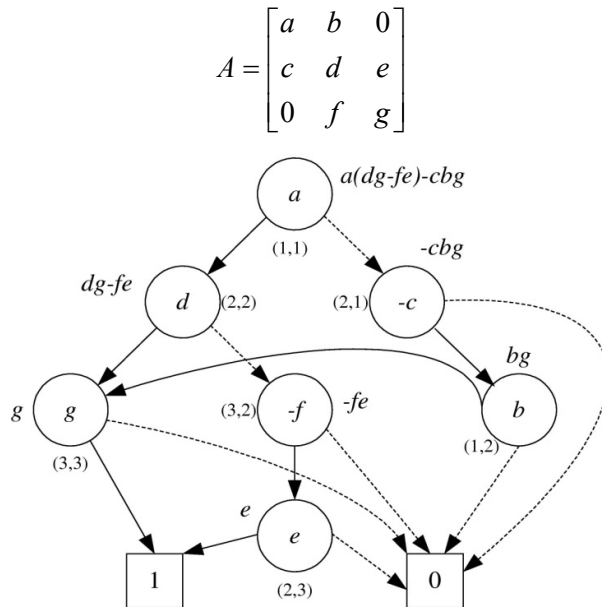


Figure 7: DDD of the matrix A.

In order to build the DDD from the matrix A , all coefficients must be sorted in descending form, for instance: $a > c > b > d > f > e > g$. A heuristic method of vertex sorting is given in [34]. Then, each vertex represents a determinant matrix D defined recursively as:

1. $D = 1$, if the vertex is pointing out to the 1 -terminal.
2. $D = 0$, if the vertex is pointing out to the 0 -terminal.
3. $D = a_i s(a_i) D_{a_i} + D_{-a_i}$, for each non-terminal vertex. Here D_{a_i} and D_{-a_i} are the determinants of the vertices that point out to the 1 -edge and 0 -edge, respectively. The sign of each vertex a_i in any 1 -path is defined as:

$$s(v) = \prod \text{sign}(r(x) - r(v)) \text{sign}(c(x) - c(v)) \quad (33)$$

where $r(x)$ and $c(x)$ are the indexes of the row and column associated to each coefficient of matrix A . The sign function is defined as:

$$\text{sign}(u) = \begin{cases} 1 & \text{for } u > 0 \\ -1 & \text{for } u < 0 \end{cases} \quad (34)$$

For the case that v has an edge pointing to the 1 -terminal vertex, then $s(v) = 1$. For instance, let us consider the vertex c with its indexes given by: $r(v) = 2$ and $c(v) = 1$. The indexes of the following vertex in the 1 -edge, labeled as b are given by: $r(x) = 1$ and $c(x) = 2$. By applying (33) and (34), the negative sign of the vertex c is obtained. On the other hand, a 1 -path in a DDD is defined as a path from root vertex to the 1 -terminal, considering all symbolic terms along with the sign of the vertices that originate all the 1 -edges. Each 1 -path represents a product of terms represented by a 1 -edge whereas the addition operation is represented by a 0 -edge. For instance, one can see that there are three product terms in **Fig. 7**, given by: $-adg$, $-aef$, and $-bcg$; they are the product terms of the determinant of the matrix A .

Despite the fact that the determinant of a matrix can efficiently be represented by DDDs, it is still necessary, however, to expand each vertex in the DDD, because each coefficient in the admittance matrix is often an s polynomial composed with different symbols. In [35], the generation of symbolic transfer functions in the form of polynomials, as shown by (3), has been introduced. This approach called s -expanded DDDs is based upon multiple-root DDDs in order to represent the determinant and its cofactors. Let us substitute each coefficient of the matrix A , shown in **Fig. 7**, by an s polynomial. Thus, the matrix A is rewritten as shown in **Fig. 8**.

Therefore, each vertex of the DDD, shown in **Fig. 7**, must first be sorted, expanded and, then, the s -variables extracted [35]. Thus, each power of s becomes a root of the multiple-root DDDs, as shown in **Fig. 8**. This technique expands all s polynomials of the original determinant in a DDD and therefore, symbolic transfer functions can be generated by finding all the 1 -paths for each s -root.

As has been reported in [35], a disadvantage of DDDs and s -expanded DDDs methods, that are based on the Laplace expansion method, is that the original matrix suffers from term cancellations. Thus, for each vertex generated in the s -expanded

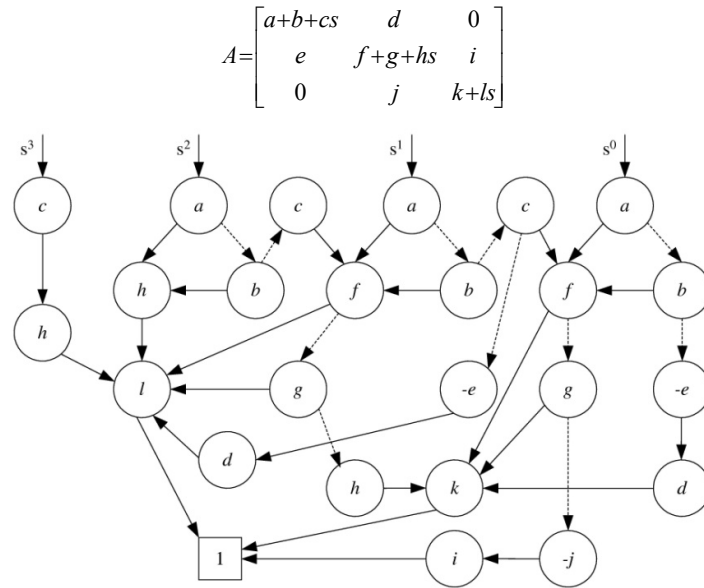


Figure 8: S-expanded DDD of the matrix A .

DDD's method, cancelling terms are indirectly introduced. In order to improve the generation of dominant terms, cancelling terms should be removed. From the admittance matrix A , one can see that there are patterns related with cancelling terms, as shown in Fig. 9.

$$\begin{matrix} l & k \\ i \begin{bmatrix} p & -p \\ -p & p \end{bmatrix} & i \begin{bmatrix} p & -q \\ -p & q \end{bmatrix} & i \begin{bmatrix} p & -p \\ -q & q \end{bmatrix} \end{matrix}$$

Figure 9: Patterns that generate cancelling terms in the admittance matrix.

An efficient algorithm to remove the patterns shown in Fig. 9 has been introduced in [36]. In this manner, the s -expanded DDD shown in Fig. 8 can be reduced as depicted in Fig. 10.

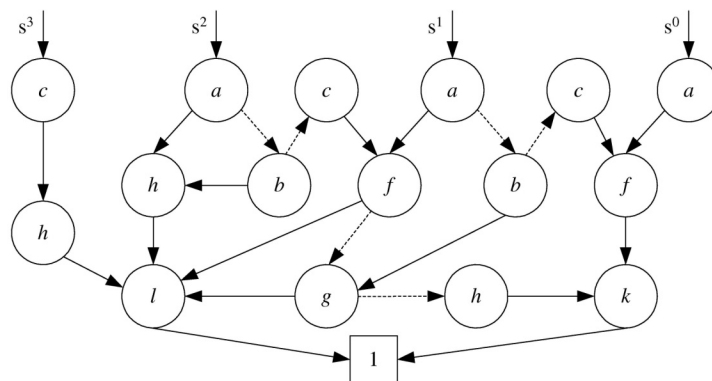


Figure 10: S-expanded multiroot DDD without cancelling terms.

In order to generate only the dominant terms by using DDDs, several algorithms based on Dynamic Programming (DP) have been proposed [37, 39]. The main idea is

to compute partial dominant terms for each DDD vertex pointing to a *l*-edge. Let us suppose that there are n vertices in a path, from root vertex to the *l*-terminal. Each vertex is composed of an s polynomial; therefore, each coefficient is first checked in order to avoid generating terms more than twice. If it does not exist, then the largest term is computed and stored in a term-list by visiting all the DDD vertices linked to *l*-edge, where its numerical value also is computed.

However, the method described above is not applicable to s -expanded DDDs, because there are vertices with incoming *l*-edges and *l*-edges. It is worth mentioning that it is necessary to duplicate some vertices in order to apply the method described above, as has been described in [37-39]. To solve this drawback, another DP-based algorithm has also been proposed in [39]. This algorithm is based on finding the shortest path in edge-weighted DDDs graphs by using the following edge weight:

1. *l*-edge with weight 0.
2. *l*-edge with weight $-\log|a_i|$

Here $|a_i|$ is the numerical value of the DDD vertex a_i of the *l*-edge. As a consequence the total weight of a path is given as:

$$w = -(\log(a_1) + \log(a_2) + \dots + \log(a_n)) \quad (35)$$

Once a shortest path has been found in a DDD, it is subtracted in a similar form as s -variables [35] and, then, the next shortest path is searched. A disadvantage of this approach is that one needs to visit each vertex in a DDD graph for each shortest path. In response to this drawback, a reverse DDD graph-based new algorithm has also been introduced in [39]. Basically, the root vertex, the two terminal vertices and all edges along with their directions in a normal DDD graph are reversed. Therefore, the two terminal vertices are now the root vertices and the root vertex in DDDs becomes the new terminal vertex. In the same way as normal DDD graphs, *l*-paths and path weights are defined for reverse DDD graphs. Thus, the dominant terms correspond to the shortest paths in the corresponding reverse DDD graph and they are extracted in the same way as s -variables [35]. Finally, after the first shortest path has been found, the next shortest path can be found by only visiting the newly added vertices created by the subtraction operation. From the DP algorithms described above, we can conclude that the latter algorithm is more efficient in order to generate dominant terms based on DDDs, since it can be applied to any topology of a circuit, which can be represented by a normal DDD or s -expanded DDD graphs.

4.2. Error control

Term generation must be accompanied by an appropriate error control methodology. Moreover, the error control methodology determines to a certain extent the appropriate term generation approach among the alternatives discussed above.

4.2.1. Coefficient-based approach

The basic error control methodology is a simple extension of the methods used in simplification after generation techniques discussed in Section 2. Approaches in [20-23, 26, 27] generate the P most significant terms in (3) for each network function

coefficient, $h_k(\mathbf{x})$, until the sum of the generated terms represents a given fraction of the total magnitude of the coefficient:

$$\left| h_k(\mathbf{x}_o) - \sum_{l=1}^P h_{kl}(\mathbf{x}_o) \right| < \varepsilon_k |h_k(\mathbf{x}_o)| \quad (36)$$

where $h_k(\bullet)$ represents any coefficient from the network function numerator or denominator, \mathbf{x}_o represents a design point of the circuit parameters and ε_k is a threshold error.

As shown in (36), the total magnitude of each circuit coefficient, $h_k(\mathbf{x}_o)$, must be known a priori. The same numerical interpolation technique used in the second SBG error control mechanism in Section 3.5 can be applied here.

This error control methodology is naturally associated to the first term generation methodology. If each coefficient in the network function were generated with the same threshold error, there would be no magnitude and phase deviations. However, even if the same error threshold were specified for each coefficient, the real error would be different for each one due to the discrete characteristic of the simplification process. As that real error is unpredictable a priori, the same happens for the magnitude and phase deviations, which may become really large.

A possible solution is to use (31) to back-propagate magnitude and phase errors specified by the user to individual errors of the network function coefficients. However, such back-propagation is extremely conservative and many more terms than those strictly necessary to meet the magnitude and phase error constraints are generated.

4.2.2. Sensitivity-based approach

This procedure starts with the elimination of unimportant coefficients [40]. This elimination is performed according to a ranking function that orders the contributions of the coefficients and eliminates those having the lowest contribution while the error constraints are met. To avoid the generation of too many terms in the remaining coefficients, only a part of the total error allowed by the user is applied in this step.

In a second step, the largest term of each of the remaining coefficients is generated and added to the final symbolic expression. Then, having queued the following largest terms of each coefficient, a sensitivity analysis allows deciding which coefficient is contributing the most to the circuit behavior. The largest term of this coefficient is added to the symbolic expression and the error constraints are checked. If the specifications are fulfilled, the term generation is finished; otherwise, the sensitivity analysis is again used to add a new term to the final symbolic expression. This procedure is repeated until the error specifications given by the user are met.

Using this methodology, the generated symbolic expression will be more compact than in the previous case. However, as in the first approach, to control the accuracy of the results, a dense set of frequency samples is needed to compare the evaluation of the generated expression to the exact circuit behavior. Then, the accuracy is not guaranteed between each pair of frequency samples.

For the generation of terms in decreasing order of magnitude within each coefficient two possibilities arise:

- *Use the intersection of the voltage graphic matroid and the partition matroid.* If this approach is used, the generated terms have to be checked to see if they are also valid spanning trees of the current graph. Unfortunately, for circuits with many voltage-controlled current sources (*i.e.* those circuits composed of transistors), the number of terms that are valid for the intersection problem but are not spanning trees of the current graph grows exponentially with the circuit size. Therefore, a lot of terms will be needed to generate a valid term for the symbolic expression.
- *Use of the intersection of the two graphic matroids.* In this case, the enumeration of bases is more costly but it is guaranteed that all the generated ones are valid. Bases can be generated only at single frequency values. In the approach in [41, 42], the sensitivity analysis is used to find the frequency at which each coefficient dominates the circuit behavior. At this frequency value, the generated terms in decreasing order of magnitude are expected to belong to the desired coefficient. However, except for low-order coefficients, these frequency values are usually too close, so terms belonging to adjacent coefficients are generated in the process. Of course, all the terms that do not belong to the desired coefficient have to be neglected. This, again, imposes an extra overhead on the number of generated terms to obtain a valid set of spanning trees.

4.2.3. Sampling-based approach

In [43], a set of frequency samples belonging to the frequency range of interest is used. For each of these samples, the matroid intersection algorithm involving the two graphic matroids, is executed, until the error criteria at such frequency value is fulfilled according to a relative threshold defined by the user. Joining together all the generated terms at all the frequency samples provides a symbolic expression valid at such samples.

A major disadvantage of this approach is that, pasting together all the terms generated at each frequency sample implies that the final symbolic expression will have more terms than those strictly necessary to meet the error constraints. And, the larger the number of frequency samples, the larger the number of extra (unnecessary) terms.

Also, the accuracy is not guaranteed between each pair of frequency samples. This will be guaranteed only in case that an infinite number of frequency samples are used; but this is obviously impossible. The possibility of violating the error constraints decreases (although never reaching zero) if the number of frequency samples is increased, but this also implies higher CPU time and more conservative results.

Another approach that palliates the problems of previous approaches is reported in [44]. To ensure that valid terms are always generated without needing a large number of frequency samples, thus, ensuring that the resulting symbolic expression is as compact as possible, this methodology proceeds as follows. For one frequency sample, and within the range defined by the user, the term generation based on the intersection of the two graphic matroids is executed until the errors at that frequency are within the specifications. By doing so, it is only guaranteed that the generated expression fulfils the error constraints in the selected frequency value. Thus, we need some mechanism to find

out if, within the frequency range, there are some frequency values in which error constraints are exceeded. This test is performed by the following maximum error detection algorithm. The difference between the magnitude and phase behaviors of the original circuit and those of the evaluation of the generated expression is calculated. Then, the maximum for the magnitude and phase errors are obtained by using the same methodology than in Section 3.5 and if any of these two errors are beyond the specifications, the frequency value at which the maximum of the errors is achieved is added to the existing set of frequency samples.

This procedure is performed iteratively until the error specifications are met in the complete frequency range defined by the user. A small number of iterations are usually enough.

This algorithm allows the fast generation of symbolic expressions with full accuracy. However, the accumulation of different frequency sampling points may result in a symbolic expression whose complexity is larger than strictly needed. This is because, as a consequence of the generation of symbolic terms in an additional frequency point, some terms already generated at other frequency may become obsolete. Note that this problem has already been drastically minimized by using a small number of frequency samples. However, to ensure that the complexity of the results is minimal, a post-processing step after the simplification process is applied. All the symbolic terms, generated at the different frequency samples, are queued according to their weight within each coefficient. Then, taking into account the contribution of the different coefficients in the network function to the circuit behavior, symbolic terms are iteratively neglected, starting with those having the smallest weight, while the error specifications are not exceeded.

5. Conclusion

The exponential growth of symbolic analysis results with the circuit size demands the application of approximated analysis techniques. Historically, simplification after generation techniques, *i.e.*, techniques that approximate the symbolic expressions once they have been generated, were the first ones to appear. They palliated the interpretation problem but the limitation on the analyzable circuit size still remained. This has been solved by the introduction of simplification before generation techniques, which simplify the system of circuit equations, and simplification during generation techniques, which directly generate simplified symbolic expressions. Most efficient analysis techniques are based on the two-graph method, which more recently have been modeled as matroid intersection problems, and on determinant decision diagrams. Approximated analysis techniques must include a suitable error control mechanism, which decides the appropriate approximate symbolic results for a given accuracy.

Acknowledgements

The preparation of this chapter has been partially supported by the TIC-2532 Project, funded by Consejería de Innovación, Ciencia y Empresa, Junta de Andalucía and by the Project TEC2010-14825/MIC, funded by the Spanish Ministry of Science and Innovation with support from ERDF. Dr. C. Sánchez-López thanks the support of the JAE-Doc program of CSIC, co-funded by FSE.

References

- [1] G. Wierzba, *Sspice user manual*, Version 1.0. Michigan State University, East Lansing, February 1991.
- [2] G. Gielen and W. Sansen, *Symbolic analysis for automated design of analog integrated circuits*. Boston: Kluwer Academic, 1991.
- [3] F. V. Fernández, A. Rodríguez-Vázquez, and J. L. Huertas, "Interactive AC modeling and characterization of analog circuits via symbolic analysis," *Analog Integrated Circuit and Signal Processing*, vol. 1, pp. 183–208, November 1991.
- [4] A. Liberatore, A. Luchetta, S. Manetti and M. C. Piccirilli, "A new symbolic program package for the interactive design of analog circuits" in *IEEE International Symposium on Circuits and Systems*, 1995, vol. 3, pp. 2209-2212.
- [5] G. Gielen, H. Walscharts, and W. Sansen, "ISAAC: A symbolic simulator for analog integrated circuits," *IEEE Journal of Solid-State Circuits*, vol. 24, pp. 1587–1597, December 1989.
- [6] F. V. Fernández, A. Rodríguez-Vázquez, J. D. Martín, and J. L. Huertas, "Formula approximation for flat and hierarchical symbolic analysis," *Analog Integrated Circuits and Signal Processing*, vol. 3, no. 1, pp. 43–58, Kluwer, January 1993.
- [7] R. E. Moore, *Methods and applications of interval analysis*. Studies in Applied Mathematics, Philadelphia, 1979.
- [8] J.-J. Hsu and C. Sechen, "Fully symbolic analysis of large analog integrated circuits", in *IEEE Custom Integrated Circuits Conference*, 1994, pp. 457-460.
- [9] J. J. Hsu and C. Sechen, "DC small signal symbolic analysis of large analog integrated circuits," *IEEE Transactions Circuits and Systems I*, vol. 41, no. 12, pp. 817-828, December 1994.
- [10] J. J. Hsu and C. Sechen, "Low-frequency symbolic analysis of large analog integrated circuits" in *IEEE Custom Integrated Circuits Conference*, 1993, pp. 14.7.1-14.7.5.
- [11] R. Sommer, E. Hennig, G. Droge and E. H. Horneber, "Equation-based symbolic approximation by matrix reduction with quantitative error prediction," *Alta Frequenza*, vol. 5, no. 6, pp. 317-325, November 1993.
- [12] Q. Yu and C. Sechen, "A unified approach to the approximate symbolic analysis of large analog integrated circuits," *IEEE Transactions on Circuits and Systems I*, vol. 43, no. 8, pp. 656-669, August 1996.
- [13] W. Daems, G. Gielen and W. Sansen, "Circuit complexity reduction for symbolic analysis of analog integrated circuits", in *IEEE/ACM Design Automation Conference*, 1999, pp. 958-963.
- [14] W. Daems, G. Gielen and W. Sansen, "Circuit simplification for the symbolic analysis of analog integrated circuits", *IEEE Transactions on Computer-Aided Design*, vol. 21, no. 4, pp. 395-407, April 2002.
- [15] O. Guerra, J.D. Rodríguez-García, E. Roca, F.V. Fernández and A. Rodríguez-Vázquez, "A simplification before and during generation methodology for symbolic large-circuit analysis", in *IEEE International Conference on Electronics, Circuits and Systems*, 1998, vol. 3, pp. 81-84.
- [16] O. Guerra, J. D. Rodríguez-García, E. Roca, F. V. Fernández and A. Rodríguez-Vázquez, "An accurate error control mechanism for simplification before generation algorithms", in *IEEE Design, Automation and Test in Europe Conference*, 1999, pp. 412-416.
- [17] J. D. Rodríguez-García, O. Guerra, E. Roca, F. V. Fernández and A. Rodríguez-Vázquez, "Error control in simplification before generation algorithms for symbolic analysis of large analogue circuits," *Electronic Letters*, vol. 35, no. 4, pp. 260-261, February, 1999.
- [18] J. Vlach and K. Singhal, *Computer methods for circuit analysis and design*. Van Nostrand Reinhold, 1994.
- [19] F. V. Fernández, O. Guerra, J. D. Rodríguez-García and A. Rodríguez-Vázquez, "Symbolic analysis of analog integrated circuits: the numerical reference generation problem," *IEEE Transactions on Circuits and Systems-II*, vol. 45, no. 10, pp. 1351-1361, October 1998.
- [20] F. V. Fernández, P. Wambacq, G. Gielen, A. Rodríguez-Vázquez and W. Sansen, "Symbolic analysis of large analog integrated circuits by approximation during expression generation," in *IEEE International Symposium on Circuits and Systems*, 1994, vol. CAD, pp. 25-28.
- [21] P. Wambacq, F.V. Fernández, G. Gielen and W. Sansen: "Efficient symbolic computation of approximated small-signal characteristics", in *IEEE Custom Integrated Circuits Conference*, 1994, pp. 461-464.
- [22] Q. Yu and C. Sechen, "Generation of color-constrained spanning trees with application in symbolic circuit analysis," in *Fourth Great Lakes Symposium on VLSI*, 1994, pp. 252-255.
- [23] Q. Yu and C. Sechen, "Approximate symbolic analysis of large analog integrated circuits," in *IEEE International Conference on Computer-Aided Design*, 1994, pp. 664-671.

- [24] P. M. Lin, *Symbolic network analysis*. Amsterdam: Elsevier, 1991.
- [25] S.P. Chan, *Introductory Topological analysis of electrical networks*. Holt, Rinehart and Winston, 1969.
- [26] P. Wambacq, F. V. Fernández, G. Gielen, W. Sansen and A. Rodríguez-Vázquez, "Algorithm for efficient symbolic analysis of large analogue circuits," *Electronics Letters*, pp. 1108-1109, July 1994.
- [27] P. Wambacq, F. V. Fernández, G. Gielen, W. Sansen and A. Rodríguez-Vázquez, "Efficient symbolic computation of approximated small-signal characteristics of analog integrated circuits," *IEEE Journal of Solid-State Circuits*, vol. 30, no. 3, pp. 327-330, March 1995.
- [28] H.N. Gabow, "Two algorithms for generating weighted spanning trees in order", *SIAM Journal of Computing*, vol. 6, no. 1, pp. 139-150, March 1977.
- [29] E.L. Lawler, *Combinatorial optimization: networks and matroids*. Holt, Rinehart and Winston, 1976.
- [30] Q. Yu and C. Sechen, "Efficient approximation of symbolic network functions using matroid intersection algorithms," in *IEEE International Symposium on Circuits and Systems*, 1995, pp. 2088-2091.
- [31] P. Wambacq, F. V. Fernández, G. Gielen, W. Sansen and A. Rodríguez-Vázquez, "A family of matroid intersection algorithms for the computation of approximated symbolic network functions" in *IEEE International Symposium on Circuits and Systems*, 1996, pp. 806-809.
- [32] P. Wambacq, "Symbolic analysis of large and weakly non-linear analog integrated circuits" Ph. D. Thesis, KU Leuven, 1996.
- [33] P. M. Camerini and H.W. Hamacher, "Intersection of two matroids: (condensed) border graph and ranking", *SIAM Journal Discrete Mathematics*, vol. 2, pp. 16-27, February 1989.
- [34] C.-J. Shi and X.-D. Tan, "Canonical symbolic analysis of large analog circuits with determinant decision diagrams", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, pp. 1-18, January 2000.
- [35] C.-J. Shi and X.-D. Tan, "Compact representation and efficient generation of s-expanded symbolic network functions for computer-aided analog circuit design", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 20, pp. 813-827, July 2001.
- [36] S. X.-D Tan and C.-J. Shi, "Parametric analog behavioral modeling based on cancellation-free DDDs", in *Proc. IEEE International Workshop on Behavioral Modeling and Simulation*, 2002, pp. 25-31.
- [37] W. Verhaegen and G. Gielen, "Symbolic determinant decision diagrams and their use for symbolic modeling of linear analog integrated circuits", *International Journal of Analog Integrated Circuits and Signal Processing*, vol. 31, pp. 119-130, 2002.
- [38] W. Verhaegen and G. Gielen, "Efficient DDD-based symbolic analysis of linear analog circuits", *IEEE Transactions on Circuits and Systems II: Analog and digital signal processing*, vol. 49, pp. 474-487, July 2002.
- [39] X.-D. Tan and C.-J. Shi, "Efficient approximation of symbolic expressions for analog behavioral modeling and analysis", *IEEE Transactions on Computer -Aided Design of Integrated Circuits and System*, vol. 23, pp. 907-918, June 2004.
- [40] W. Daems, W. Verhaegen, P. Wambacq, G. Gielen and W. Sansen, "Evaluation of error control strategies for the linear symbolic analysis of analog integrated circuits", *IEEE Transactions Circuits and Systems I*, vol. 46, no. 5, pp. 594-606, May 1999.
- [41] P. Dobrovolny, P. Wambacq, G. Gielen and W. Sansen, "Efficient symbolic analysis of large analog circuits using sensitivity-driven ranking of matroid intersections," in *IEEE International Symposium on Circuits and Systems*, 1998, vol. 6, pp. 29-32.
- [42] P. Wambacq, P. Dobrovolny, G. Gielen and W. Sansen, "Symbolic analysis of large analog circuits using a sensitivity-driven enumeration of common spanning trees," *IEEE Transactions Circuits and Systems II*, vol. 45, no. 10, pp. 1342-1350, Oct. 1998.
- [43] Q. Yu and C. Sechen, "Efficient approximation of symbolic network functions using matroid intersection algorithms", *IEEE Transactions on Computer-Aided Design*, vol. 16, pp. 1073-1081, October 1997.
- [44] O. Guerra, J.D. Rodríguez-García, E. Roca, F.V. Fernández and A. Rodríguez-Vázquez, "A simplification before and during generation methodology for symbolic large-circuit analysis," in *IEEE International Conference on Electronics, Circuits and Systems*, 1998, vol. 3, pp. 81-84.

CHAPTER 8**Symbolic Analysis by Determinant Decision Diagrams and Applications****Sheldon X.-D. Tan****Department of Electrical Engineering, University of California, Riverside, USA*

Abstract: Symbolic analysis traditionally suffers circuit size problems as the number of symbolic terms generated can grow exponentially with the circuit size. This problem has been partially mitigated by a graph-based approach, called Determinant Decision Diagram (DDD) [1], where the symbolic terms are implicitly represented in a graph, which has been inspired by the success of Binary Decision Diagram (BDDs) [2] as an enabling technology for industrial use of symbolic analysis and formal verification in digital logic design. DDD-based symbolic analysis enables the exact symbolic analysis of many analog circuits substantially larger than the previous methods and open new applications for symbolic analysis. DDD-based symbolic analysis still remains the most efficient symbolic analysis technique. This chapter will present basic concept of DDDs, the most efficient DDD construction method based on logic operation, s-expanded DDDs for generating s-expanded polynomials and transfer functions. We will also show how DDDs and s-expanded DDDs can be used for constructing simplified symbolic expressions.

Keywords: Symbolic analysis, determinant decision diagrams, analog circuits, modeling, simulation and analysis, binary decision diagrams, compact modeling.

1. Exact Symbolic Analysis by Determinant Decision Diagrams

It is well known that the primary difficulty in exact symbolic analysis is the exponential growth of the number of product terms with the size of circuits. In this chapter, we introduce a graph-based symbolic analysis approach, which partially mitigate the long-standing circuit-size problem for exact symbolic analysis. The graph based approach is based on the two observations on symbolic analysis of a large analog circuit: (a) the circuit matrix is sparse and (b) a symbolic expression often shares many sub-expressions. Under the assumption that all the matrix elements are distinct, each product term can be viewed as a subset of all the symbolic parameters.

Determinant decision diagrams (DDD) were introduced to represent determinants symbolically [1]. DDDs are essentially Zero-suppressed Binary Decision Diagrams (ZBDDs) — introduced originally for representing sparse subset systems [3]. A ZBDD is a variant of a Binary Decision Diagram (BDD) introduced by Akers [4] and popularized by Bryant [2]. BDDs have brought a great success to the formal verification and testing for combinational and sequential digital circuits [2, 5]. DDD representation has several advantages over both the expanded and arbitrarily nested forms of a symbolic expression. First, similar to the nested form, our representation is compact for a large class of analog circuits. A ladder-structured network can be represented by a diagram where the number of vertices in the diagram (called its *size*) is equal to the number of symbolic parameters.

As indicated by [1], the typical size of DDD is dramatically smaller than that of

*Address correspondence to Sheldon X.-D. Tan: Department of Electrical Engineering, UC, Riverside, USA; E-mail: stan@ee.ucr.edu

product terms. For instance, 5.71×10^{20} terms can be represented by a diagram with 398 vertices [1]. Second, similar to the expanded form, our representation is canonical, i.e., every determinant has a unique representation, and is amenable to symbolic manipulations.

2. DDD Representation of Symbolic Determinant

In this section, we formally introduce determinant decision diagrams to represent a symbolic matrix determinant. DDDs are actually canonical representations of matrix determinants, similar to BDDs for representing binary functions and ZBDDs for representing subset systems.

A key observation is that the circuit matrix is sparse and that many times, a symbolic expression may share many sub-expressions. For example, consider the following determinant

$$\det(M) = \begin{vmatrix} a & b & 0 & 0 \\ c & d & e & 0 \\ 0 & f & g & h \\ 0 & 0 & i & j \end{vmatrix} = adgj - adhi - aefj - bcgj + cbih \quad (1)$$

We note that sub-terms ad , gj , and hi appear in several product terms, and each product term involves a subset (four) out of ten symbolic parameters. Therefore, we view each symbolic product term as a subset, and use a zero-suppressed binary decision diagram (ZBDD) [3], to represent the subset system composed of all the subsets each corresponding to a product term. **Fig. 1** illustrates the corresponding

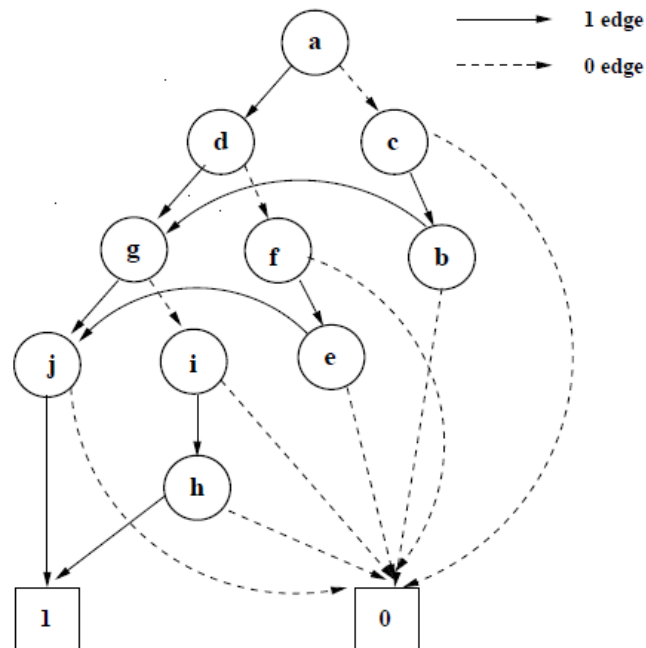


Figure 1: A ZBDD representing $\{adgj, adhi, aefj, bcgj, cbih\}$ under ordering $a > c > b > d > f > e > g > i > h > j$. (Copyright © 2000, Re-printed with permission from [1])

ZBDD representing all the subsets involved in $\det(M)$ under ordering $a > c > b > d > f > e > g > i > h > j$. It can be seen that sub-terms ad , gj , and ih have been shared in the ZBDD representation.

Following directly from the properties of ZBDDs, we have the following observations. First, given a fixed order of symbolic parameters, all the subsets in a symbolic determinant can be represented uniquely by a ZBDD. Second, every 1-path (1-path is a path from the root to the 1 terminal) in the ZBDD corresponds to a product term, and the number of 1-edges (1-edge is the true edge in the original BDD) in any 1-path is n . The total number of 1-paths is equal to the number of product terms in a symbolic determinant.

We can view the resulting ZBDD as a graphical representation of the recursive application of the determinant expansion with the expansion order $a, c, b, d, f, e, g, i, h, j$. Each vertex is labeled with the matrix entry with respect to which the determinant is expanded, and it represents all the subsets contained in the corresponding sub-matrix determinant. The 1-edge points to the vertex representing all the subsets contained in the COFACTOR operation (defined later) of the current expansion, and 0-edge points to the vertex representing all the subsets contained in the REMAINDER operation (defined later).

To embed the signs of the product terms of a symbolic determinant into its corresponding ZBDD we associate each vertex v with a sign, $s(v)$, defined as follows:

1. Let $P(v)$ be the set of ZBDD vertices that originate the 1-edges in any 1-path rooted at v , then

$$s(v) = \prod_{x \in P(v)} \text{sign}(r(x) - r(v)) \text{sign}(c(x) - c(v)), \quad (2)$$

where $r(x)$ and $c(x)$ refer to the absolute row and column indices of vertex x in the original matrix, and u is an integer so that

$$\text{sign}(u) = \begin{cases} 1, & u > 0 \\ -1, & u < 0 \end{cases}$$

2. If v has an edge pointing to the 1-terminal vertex, then $s(v) = +1$.

This is called the *sign rule*. For example, in **Fig. 2**, shown beside each vertex are the row and column indices of that vertex in the original matrix, as well as the sign of that vertex obtained by using the sign rule above. For the sign rule, we have the following result:

Theorem 1 *The sign of a DDD vertex v , $s(v)$, is uniquely determined by (2), and the product of all the signs in a path is exactly the sign of the corresponding product term.*

For example, consider the 1-path $acbgih$ in **Fig. 2**. The vertices that originate all the 1-edges are c, b, i, h , their corresponding signs are $-, +, -$ and $+$, respectively. Their product is $+$. This is the sign of the symbolic product term $cbih$.

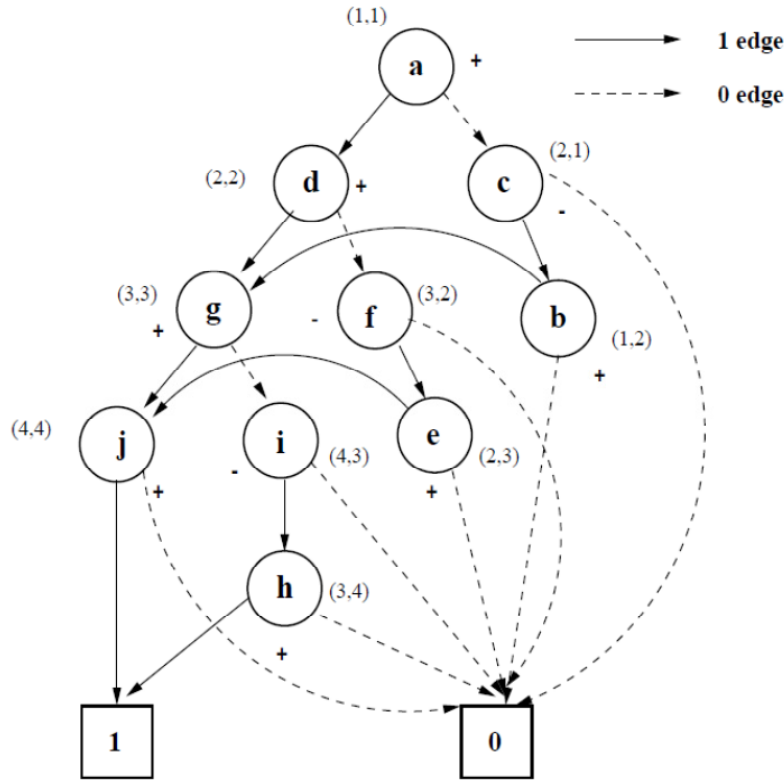


Figure 2: A signed ZBDD for representing symbolic terms. (Copyright © 2000, Re-printed with permission from [1])

With ZBDD and the sign rule as two foundations, we are now ready to introduce formally our representation of a symbolic determinant. Let \mathbf{A} be an $n \times n$ sparse matrix with a set of distinct m symbolic parameters $\{a_1, \dots, a_m\}$, where $1 \leq m \leq n^2$. Each symbolic parameter a_i is associated with a unique pair $r(a_i)$ and $c(a_i)$, which denote, respectively, the row index and column index of a_i . A determinant decision diagram is a signed, rooted, directed acyclic graph with two terminal vertices, namely the 0-terminal vertex and the 1-terminal vertex. Each non-terminal vertex a_i is associated with a sign, $s(a_i)$, determined by the sign rule defined by (2). It has two outgoing edges, called 1-edge and 0-edge, pointing, respectively, to Da_i and $D\bar{a}_i$. A determinant decision graph having root vertex a_i denotes a matrix determinant D defined recursively as

1. If a_i is the 1-terminal vertex, then $D = 1$.
2. If a_i is the 0-terminal vertex, then $D = 0$.
3. If a_i is a nonterminal vertex, then $D = a_i s(a_i) Da_i + D\bar{a}_i$.

Here $s(a_i)Da_i$ is the COFACTOR of D with respect to a_i , Da_i is the MINOR of D with respect to a_i , $D\bar{a}_i$ is the REMAINDER of D with respect to a_i , and operations are algebraic multiplications and additions. For example, **Fig. 3** shows the DDD representation of $\det(M)$ under ordering $a > c > b > d > f > e > g > i > h > j$.

To enforce the uniqueness and compactness of the DDD representation, the three rules of ZBDDs, namely, zero-suppression, ordered, and shared are adopted. This leads to DDDs having the following properties:

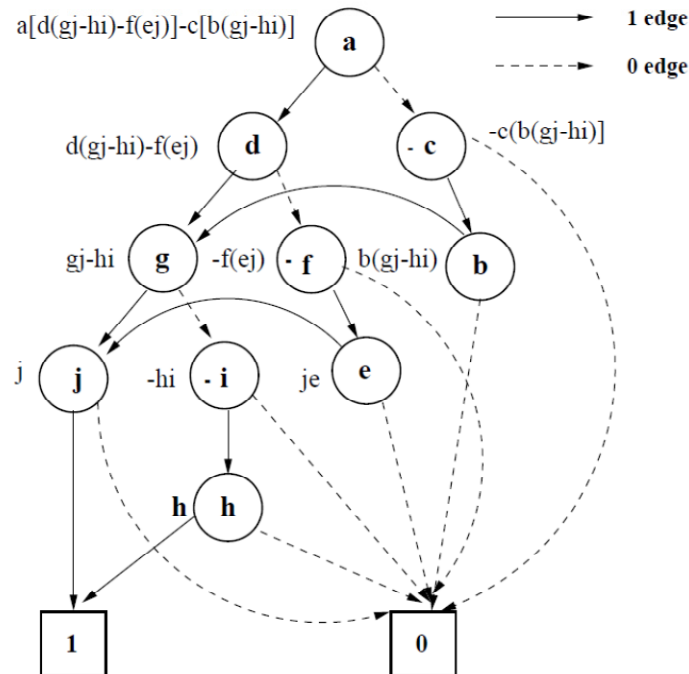


Figure 3: A determinant decision diagram for matrix M . (Copyright © 2000, Re-printed with permission from [1])

- Every 1-path from the root corresponds to a product term in the fully expanded symbolic expression. It contains exactly n 1-edges. The number of 1-paths is equal to the number of product terms.

- For any determinant D , there is a unique DDD representation under a given vertex ordering.

We use $|DDD|$ to denote the size of a DDD, i.e., the number of vertices in the DDD.

A key problem in many decision diagram applications is how to select a vertex ordering, since the size of the resulting decision diagram strongly depends on the chosen ordering. A efficient DDD vertex ordering heuristic has been developed, which can lead to the optimal vertex ordering for a class of circuit matrices, called band matrices [1, 6].

3. Manipulation of Determinant Decision Diagrams

In this section, we show that, using determinant decision diagrams, algorithms needed for symbolic analysis and its applications can be performed with the time complexity proportional to the size of the diagrams being manipulated, not the number of 1-paths in the diagrams, i.e., product terms in the symbolic expressions. Hence, as long as the determinants of interest can be represented by reasonably small graphs, our algorithms are quite efficient.

A basic set of operations on matrix determinants is summarized in **Table 1**. Most operations are simple extensions of subset operations introduced by Minato on ZBDDs [3]. These few basic operations can be used directly and/or combined to

perform a wide variety of operations needed for symbolic analysis. In this section, we first describe these operations, and then use an example to illustrate the main ideas of these operations and how they can be applied to compute network function sensitivities—a key operation needed in optimization and testability analysis. We also show that the generation of significant product terms can be cast as the k -shortest path problem in a DDD and solved elegantly in time $O(k \cdot |DDD|)$.

3.1. Implementation of basic operation

We summarize the implementation of these operations in **Fig. 4**. For the clarity of the description, the steps for computing the signs associated with DDD vertices, using the sign rule defined in Section 2, are not shown.

As the basis of implementation, we employ two techniques originally developed by Brace, Rudell and Bryant for efficiently implementing decision diagrams [7]. First, a basic procedure `GETVERTEX(top, D_1, D_0)` is to generate (or copy) a vertex for a symbol top and two sub-graphs D_1 and D_0 . In the procedure, a hash table is used to keep each vertex unique, and vertex elimination and sharing are managed mainly by `GETVERTEX`. With `GETVERTEX`, all the operations for DDDs we need are described in **Fig. 4**. Second, similar to conventional BDDs, we use a cache to remember the results of recent operations, and refer to the cache for every recursive call. In this way, we can avoid duplicate executions for equivalent sub-graphs. This enables us to execute these operations in a time linearly proportional to the size of a graph.

Evaluation: Given a determinant decision diagram pointed to by D and a set of numerical values for all the symbolic parameters, `EVALUATE(D)` computes the numerical value of the corresponding matrix determinant. `EVALUATE(D)` naturally exploits sub-expression sharing in a symbolic expression, and has time complexity linear in the size of the diagram.

Cofactor and Derivative: `COFACTOR(D, p)` is to compute the COFACTOR of a symbolic determinant D represented by a DDD with respect to symbolic parameter p . It is exactly the derivative of D with respect to p . COFACTOR is perhaps the most important operation in symbolic analysis of analog circuits. For example, the network functions can be obtained by first computing some COFACTORS, and then combining these COFACTORS according to some rules (Cramer's rule).

Table 1: Summary of Basic Operations

Determinant operation	Result	Subset operations
<code>VERTEXONE()</code>	return 1	<code>Base()</code>
<code>VERTEXZERO()</code>	return 0	<code>Empty()</code>
<code>COFACTOR(D, s)</code>	return the COFACTOR of D wrt s	<code>Subset1(D, s)</code>
<code>REMAINDER(D, s)</code>	return the REMAINDER of D wrt s	<code>Subset0(D, s)</code>
<code>MULTIPLY(D, s)</code>	return $s \times D$	<code>Change(D, s)</code>
<code>SUBTRACT(D, P)</code>	return $D - P$	<code>Diff(D, P)</code>
<code>UNION(D, P)</code>	return $D + P$	<code>Union(D, P)</code>
<code>EVALUATE(D)</code>	return the numerical value of D	--

```

COFACTOR( $D, s$ )
1  if ( $D.top < s$ ) return VERTEXZERO()
2  if ( $D.top = s$ ) return  $D_1$ 
3  if ( $D.top > s$ ) return GETVERTEX ( $D.top$ ,
    COFACTOR( $D_0, s$ ), COFACTOR( $D_1, s$ ))

REMAINDER( $D, s$ )
1  if ( $D.top < s$ ) return  $D$ 
2  if ( $D.top = s$ ) return  $D_0$ 
3  if ( $D.top > s$ ) return GETVERTEX ( $D.top$ ,
    REMAINDER( $D_0, s$ ), REMAINDER( $D_1, s$ ))

MULTIPLY( $D, s$ )
1  if ( $D.top < s$ ) return GETVERTEX( $s, 0, D$ )
2  if ( $D.top = s$ ) return GETVERTEX( $s, D_1, D_0$ )
3  if ( $D.top > s$ ) return GETVERTEX( $D.top$ ,
    MULTIPLY( $D_0, s$ ), MULTIPLY( $D_1, s$ ))

SUBTRACT( $D, P$ )
1  if ( $D = 0$ ) return VERTEXZERO()
2  if ( $P = 0$ ) return  $D$ 
3  if ( $D = P$ ) return VERTEXZERO()
4  if ( $D.top > P.top$ ) return GETVERTEX( $D.top$ , SUBTRACT( $D_0, P$ ),  $D_1$ )
5  if ( $D.top < P.top$ ) return SUBTRACT( $D, P_0$ )
6  if ( $D.top = P.top$ )
    return GETVERTEX( $D.top$ , SUBTRACT( $D_0, P_0$ ), SUBTRACT( $D_1$ ,
 $P_1$ ))

UNION( $D, P$ )
1  if ( $D = 0$ ) return  $P$ 
2  if ( $P = 0$ ) return  $D$ 
3  if ( $D = P$ ) return  $P$ 
4  if ( $D.top > P.top$ ) return GETVERTEX( $D.top$ , UNION( $D_0, P$ ),  $D_1$ )
5  if ( $D.top < P.top$ ) return GETVERTEX( $P.top$ , UNION( $D, P_0$ ),  $P_1$ )
6  if ( $D.top = P.top$ )
    return GETVERTEX( $D.top$ , UNION( $D_0, P_0$ ), UNION( $D_1, P_1$ ))

EVALUATE( $D$ )
1  if ( $D = 0$ ) return 0
2  if ( $D = 1$ ) return 1
3  return EVALUATE( $D_0$ ) +  $s(D) * D.top * EVALUATE(D_1)$ 

```

Figure 4: Implementation of basic operations for symbolic analysis (Copyright © 2000, Re-printed with permission from [1])

4. DDD Construction by Logic Operations

One important problem for DDD-based symbolic analysis is to generate the DDD graphs for a given determinant. One simple way to construct the DDD is by means of Laplace expansion and building the DDD graphs by means of basic DDD operations shown in **Table 1** as done in [1, 6]. However, such explicit and sequential generation

method can lead to exponential construction time even the final DDD sizes do not grow exponentially [8].

In this section, we look at the generation side of the symbolic analysis problem. We present a novel approach to generating all the symbolic expressions implicitly and simultaneously. The new approach is inspired by the symbolic approach to pointer analysis for compilation optimization [9] where logic functions are used to construct the symbolic invocation graphs. The main idea of the new approach is that the symbolic expression generation is viewed as a logic circuit synthesis process, and we design a logic circuit that can detect whether or not a symbolic term is a valid product term from a determinant. The logic circuit, which is essentially a Boolean function, can be represented by binary decision diagrams (BDDs). BDDs are then trivially transformed into zero-suppressed binary decision diagrams (ZBDDs), which are essentially a DDD representation of the determinant.

The most important advantage of the new approach over existing ones is that the time complexity is no longer tied to the number of product terms but depends on the implicit representation of designed logic during the entire construction process. This makes the symbolic analysis problem much more tractable as sizes of BDD/DDD graphs typically grow very slowly with circuit sizes given a good variable ordering. The new symbolic analysis method shows an inherent relationship between circuit simulation and logic synthesis for the first time.

4.1. Terms-detecting logic for a determinant

The DDD graph is introduced to represent a determinant. It essentially represents all the product terms in the determinant. In a DDD graph, each product term corresponds to a 1-path from the root vertex to the 1-terminal. If we view a DDD graph as a BDD graph, where each symbol in a product term takes true Boolean value, all the other symbols take false Boolean value, then the DDD essentially represents the logic that detects if a given symbolic term is a product term in the determinant, as a valid product term always corresponds to an 1-path, and thus satisfies the logic.

This motivates us to generate the DDD graph by constructing a logic circuit which is able to detect if a given product term is a valid one from the determinant. This turns out to be an easy design problem. Indeed, from the definition of determinant [10], we can design a logic to check whether the rows and columns of all the elements in a symbolic term cover every row and column of the matrix exactly once.

Fig. 5 shows a portion of the logic schematic for checking whether a given product term is valid from an $n \times n$ matrix. We simply compare the row/column index of each nonzero element in this product term with the index of each row/column and examine if each row/column index appears exactly once.

The logic in **Fig. 5** checks for row 1 (encoded as 001 since 3 bit binary coding is used in this example). $a_{11}, a_{21} \dots a_{m1}$ are the elements in the product term to be checked, 001, 010..., $b_2 b_1 b_0$ are the binary codes for all row indices in the matrix. T_1 is true only when one of its inputs is true, ensuring that exactly one nonzero element is in row 1. Comparators C_1 to C_N compare the row index of each nonzero element with the row index of row 1. (N is the total number of nonzero elements in the matrix). The *AND* gate in the last stage makes sure that all the row indices of the matrix are present in the product term. The resulted Boolean function for the row index legality check is f_{row} .

We can do the same for the column index legality check where each nonzero element is compared with the column index of each column. The resulting logic function for

column index legality check is f_{col} . Since both row and column legality conditions must be satisfied to make a valid product term, the final logic is the conjuncture (AND operation) of two logic functions:

$$f_{det} = f_{row} \cap f_{col} = f_{row}f_{col} \tag{3}$$

where \cap operation is the logic AND operation. We may also write the $f_{row} \cap f_{col}$ as $f_{row}f_{col}$ in the sequel. The resulting logic f_{det} is the Boolean logic we are looking for.

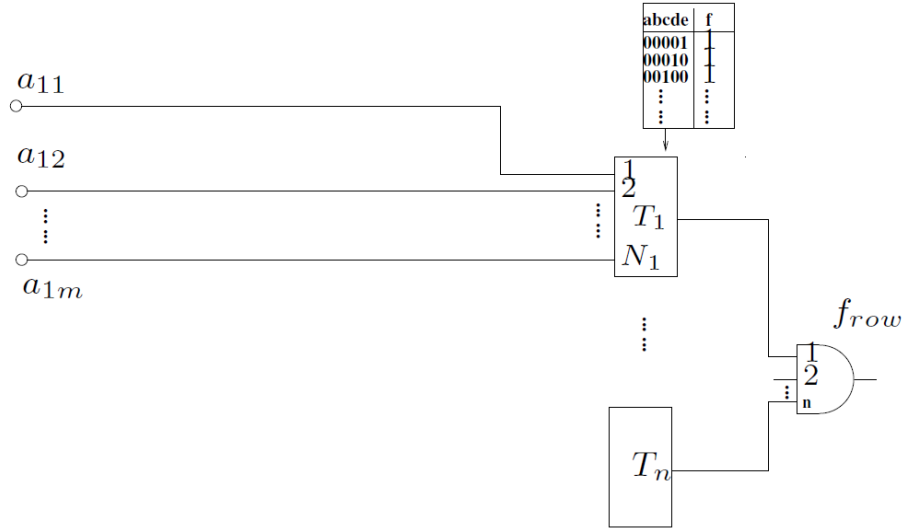


Figure 5: The logic circuit for detecting a valid product term from a determinant. (Copyright © 2006, Re-printed with permission from [22])

4.2. Logic operation based DDD construction algorithm

In this section, we show that the logic circuit shown in **Fig. 5** can be further simplified and the DDD construction can be performed efficiently by a number of simple logic operations.

4.2.1. Efficient BDD construction for the determinant detecting logic

For the determinant detecting logic circuit in **Fig. 5**, we observe that if the nonzero element a_{ij} is not in row 1, then the comparison result will always be 0 (i.e. C_i is always 0). On the other hand, if the a_{ij} is in row 1, the C_i will be a_{ij} where a_{ij} is a Boolean variable. Suppose that row 1 has three nonzero elements a_{11}, a_{12} and a_{13} then we have

$$T_1 = a_{11}\bar{a}_{12}\bar{a}_{13} + \bar{a}_{11}a_{12}\bar{a}_{13} + \bar{a}_{11}\bar{a}_{12}a_{13} \tag{4}$$

where “+” is the OR operation. As a result, we conclude that each nonzero element in a row i will generate a product term for each row’s uniqueness checking function T_i . In the product term of each nonzero element, the corresponding nonzero element will take true Boolean value while the rest nonzero elements in the same row will take

false Boolean value. So every nonzero element in a determinant will generate one product term for constructing f_{row} .

For a $n \times n$ matrix, the row legality checking function f_{row} becomes:

$$f_{row} = T_1 \cap T_2 \dots \cap T_n \quad (5)$$

We do the same for generating the column legality check function f_{col} where every nonzero element generates one product term also for f_{col} . We can directly build those product terms from a determinant by inspection, which simplifies the BDD construction considerably. Theoretically, we have

Theorem 2 *A product term is a valid one product term of a given matrix determinant $det(A)$ if and only if (after the product term is transformed into a Boolean expression), it satisfies the Boolean function $f_{det}(A) = (f_{row} \cap f_{col})$. f_{row} and f_{col} are defined above for determinant $det(A)$.*

In the following, we illustrate such construction using a simple 2×2 determinant $det(A_{2 \times 2})$ as shown below:

$$det(A_{2 \times 2}) = \begin{vmatrix} a_{11} & 0 \\ a_{21} & a_{22} \end{vmatrix} = a_{11} a_{12}$$

Determinant $det(A_{2 \times 2})$ only has one product term $a_{11}a_{22}$. We now show how this product term can be generated by using the aforementioned logic circuit. First, we construct row legality check Boolean function f_{row} . For row 1, we have $T_{r,1} = a_{11}$. For row 2, we have $T_{r,2} = a_{21}\bar{a}_{22}$. As a result, f_{row} becomes

$$f_{row} = T_{r,1} \cap T_{r,2} = a_{11}(a_{21}\bar{a}_{22} + \bar{a}_{21}a_{22})$$

Then we construct column legality check Boolean function f_{col} . For column 1, we have a_{11} and $T_{c,1} = a_{11}\bar{a}_{21} + a_{11}a_{21}$. For the column 2, we have $T_{c,2} = a_{22}$. As a result, f_{col} becomes

$$f_{col} = T_{c,1} \cap T_{c,2} = a_{22}(a_{11}\bar{a}_{21} + a_{11}a_{21})$$

The final BDD representing all the product terms from $det(A_{2 \times 2})$ is

$$\begin{aligned} f_{det(A_{2 \times 2})} &= f_{row} \cap f_{col} \\ &= (a_{11}(a_{21}\bar{a}_{22} + \bar{a}_{21}a_{22})) (a_{22}(a_{11}\bar{a}_{21} + a_{11}a_{21})) \\ &= a_{11}a_{22}\bar{a}_{21}. \end{aligned}$$

Boolean expression $a_{11}a_{22}\bar{a}_{21}$ actually is exactly the BDD representation of the valid product term $a_{11}a_{22}$ as \bar{a}_{21} will be suppressed when the BDD graph is transformed

into the ZBDD graph (DDD). Note that the sign of each node in the DDD will be computed when the DDD is constructed from the corresponding BDD.

4.2.2. New construction algorithm

In this subsection, we outline the new BDD construction algorithm for the determinant detecting logic shown in **Fig. 5**. For a nonzero element a_{ij} at row i , let $P_r(a_{ik})$ designate the product term where a_{ik} takes true Boolean value while the rest nonzero elements in row i take false Boolean value, a_{il} , $l \neq k$. The same is true for product term $P_c(a_{jk})$ for a nonzero element a_{kj} in a column j . Then the BDD construction algorithm is given in **Fig. 6**.

```

BDDCONSTRUCTBYLOGIC (A) {
  For each row  $i$  in matrix  $A$ 
     $T_{r,i} = \sum_{k=1}^n P_r(a_{ik})$ 
     $f_{row} = f_{row} \cap T_{r,i}$ ;
  For each column  $j$  in matrix  $A$ 
     $T_{c,j} = \sum_{k=1}^n P_c(a_{kj})$ ;
     $f_{col} = f_{col} \cap T_{c,j}$ ;
   $f_{det(A)} = f_{row} \cap f_{col}$ ;
  return  $f_{det(A)}$ ;
}

```

Figure 6: BDD construction algorithm for the determinant detecting logic. (Copyright © 2006, Reprinted with permission from [22])

It can be seen that the BDD construction boils down to a number of *AND* operations. We just *AND* all $T_{x,i}$ from every row and column. Once the BDD is constructed, the DDD is obtained by suppressing all the vertices with their 1-edge pointing the 0-terminal. This can be done trivially by one traversal of the BDD graph.

4.3. Logic synthesis perspective

Although the DDD construction process can be simplified into a sequence of simple logic operations, we stress that the main idea of the new method is still based on the logic synthesis concept: we generate the desired symbolic expression in terms of DDD graphs (for a determinant, its COFACTOR) by constructing proper logic circuits. So we need to first design the circuits as shown in **Fig. 5**. Once those logic circuits are designed, we can represent such circuits in terms of BDDs. In this section, we mainly show that such a transformation process can be further simplified into a number of simple Boolean operations for the construction of DDDs.

4.4. Time complexity analysis

The time complexity of the proposed method can roughly be related to the general time complexity of BDD operations, which are proportional to the sizes of the resulting BDD graphs of two operations. But the sizes of the BDD graph highly depends on the variable ordering, which in the best case has linear time complexity and in the worst case (parity functions) will still have exponential growth with the size of the number of Boolean variables (circuit sizes in our case). But many practical circuits have very small BDD sizes compared to the number of their minimum product terms, which makes BDD methods very useful for many logic synthesis and verification applications. In our BDD/DDD based symbolic analysis, we see the similar time complexity. But from the symbolic analysis perspective, such time complexity is significant as the time complexity is no longer related to the number of product terms any more. Instead it depends on the size of BDDs representing the product terms at all the time.

5. s-Expanded Determinant Decision Diagrams

For many symbolic analysis applications, DDD representation is still inadequate. These applications commonly require symbolic expressions to be represented in the so-called fully expanded form in s or in the s -expanded form. For an $n \times n$ circuit matrix $A(s)$ with its entries being the linear function in the complex frequency s , its determinant, $\det(A(s))$, can be written into an s -expanded polynomial of degree n :

$$\det(A(s)) = a_n s^n + a_{n-1} s^{n-1} + \dots + a_0 \quad (6)$$

As a result, the same linear(ized) circuit transfer function $H(s)$ can be written in the following s -expanded form:

$$H(s) = \frac{\sum f_i(p_1, p_2, \dots, p_m) s^i}{\sum g_i(p_1, p_2, \dots, p_m) s^j},$$

where $f_i(p_1, p_2, \dots, p_m)$ and $g_i(p_1, p_2, \dots, p_m)$ are symbolic polynomials that do not contain the complex variable s . Despite the usefulness of s -expanded symbolic expressions, no efficient derivation method exists. The difficulty is still rooted in the huge number of s -expanded product terms that are far beyond the capabilities of symbolic analyzers using traditional methods. Although the numerical interpolation method can generate s -expanded expressions, only the complex frequency s is kept as a symbol. This method also suffers the numerical problem due to the ill-conditioned equations for solving for numerical coefficients, and thus has limited applications.

We present an efficient algorithm of constructing an s -expanded DDD from an original DDD. If the maximum number of admittance parameters in an entry of a circuit matrix is bounded (true for most practical analog circuits), we prove that both the size of the resulting s -expanded DDD and the time complexity of the construction algorithm is $O(m|D|)$, where m is the highest power of s in the s -expanded polynomial of the determinant of the circuit matrix and $|D|$ is the size of the original DDD D representing the determinant. Experimental results indicate that the number of DDD vertices used can be many orders-of-magnitudes less than that of product terms represented by the DDDs.

With s -expanded expressions, approximation on symbolic transfer functions can be performed very efficiently. In addition, symbolic poles and zeros partial symbolic analysis and symbolic circuit-level noise analysis and modeling method can be performed [6, 11].

5.1. s -Expanded symbolic representation

In this section, we introduce the concept of s -expanded determinant decision diagrams. Instead of presenting the concept in a formal way, we illustrate it through a circuit example.

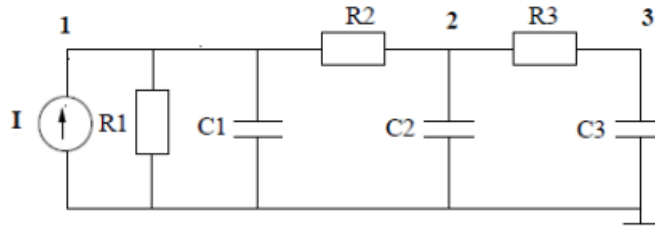


Figure 7: An example circuit. (Copyright © 2001, Re-printed with permission from [12])

Consider a simple circuit given in **Fig. 7**. By using the nodal formulation, its circuit matrix can be written as

$$\begin{matrix} v_1 \\ v_2 \\ v_3 \end{matrix} \begin{bmatrix} \frac{1}{R_1} + sC_1 + \frac{1}{R_2} & -\frac{1}{R_2} & 0 \\ -\frac{1}{R_2} & \frac{1}{R_2} + sC_2 + \frac{1}{R_3} & -\frac{1}{R_3} \\ 0 & -\frac{1}{R_3} & \frac{1}{R_3} + sC_3 \end{bmatrix}$$

In modified nodal analysis formulation, the admittance of each circuit or lumped circuit parameter, p_i , arrives in the circuit matrix in one of three forms— g_i , $c_i s$ and $1/(l_i s)$ —for the admittance of resistances and capacitances and inductances, respectively. To construct DDDs, we need to associate a label with each entry of a circuit matrix. We call this procedure labeling scheme.

Instead of labeling one symbol for each matrix entry, we label each admittance parameter in the entries of the circuit matrix when deriving the s -expanded DDDs. Depending on how the circuit parameters are labeled, an s -expanded DDD comes in two forms:

1. In the first labeling scheme, all the circuit parameters in an entry of circuit matrix are first lumped together according to their admittance type, and each lumped admittance parameter is then represented by a unique symbol.
2. In the second labeling scheme, we label each admittance of the circuit parameters by a unique symbol.

Obviously the second labeling scheme will generate more product terms than the first. The selection of labeling schemes depends on the applications of symbolic analysis. In this chapter, we present both labeling schemes along with their implementations.

By the first labeling scheme, we can rewrite the circuit matrix of the example circuit as follows:

$$\begin{bmatrix} a + bs & c & 0 \\ d & e + fs & g \\ 0 & h & i + js \end{bmatrix}$$

where $a = \frac{1}{R_1} + \frac{1}{R_2}$, $b = C_1$, $c = d = -\frac{1}{R_2}$, $e = \frac{1}{R_2} + \frac{1}{R_3}$, $f = C_2$, $g = h = -\frac{1}{R_3}$, $i = \frac{1}{R_3}$ and $j = C_3$.

By using the second labeling scheme, the circuit matrix can be rewritten as follows:

$$\begin{bmatrix} a + b + cs & d & 0 \\ d & f + g + hs & i \\ 0 & j & k + ls \end{bmatrix},$$

where $a = \frac{1}{R_1}$, $b = f = \frac{1}{R_2}$, $d = e = -\frac{1}{R_2}$, $g = k = \frac{1}{R_3}$, $i = j = -\frac{1}{R_3}$, $l = \frac{1}{R_3}$, $c = C_1$, $h = C_2$, $l = C_3$.

We first consider the original DDD representation shown in **Fig. 8** of the circuit matrix. Each DDD vertex is labeled using the first labeling scheme.

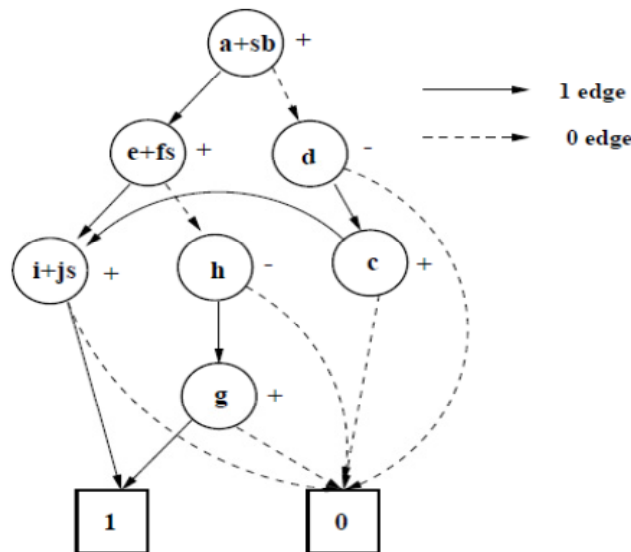


Figure 8: Complex DDD for a matrix determinant. (adapted from [21])

By the definition of DDDs, each 1-path in a DDD corresponds to a product term in the determinant that the DDD represents. In this example, there are three 1-paths, and thus three product terms:

$$\begin{aligned} &(a + sb)(e + fs)(i + js), \\ &(a + sb)(-h)(g), \\ &(-d)(c)(i + js). \end{aligned}$$

We now consider how to expand a symbolic expression into an s-expanded one and represent the expanded product terms by a new DDD structure. Expanding the three product terms, we have

$$(a + sb)(e + fs)(i + js) \rightarrow \begin{cases} +aeis^0 & +aejs^0 \\ +afis^1 & +beis^1 \\ +afjs^2 & +bejs^2 \\ +bfis^2 & +bfis^3 \end{cases}$$

$$(a + sb)(-h)(g) \rightarrow \begin{cases} -ahgs^0 \\ -bhgs^1 \end{cases}$$

$$(-d)(c)(i + js) \rightarrow \begin{cases} -dcis^0 \\ -dcjs^1 \end{cases}$$

We can easily represent these product terms using a multi-rooted DDD structure as shown in Fig. 9. The new DDD has four roots and each DDD root represents a symbolic expression of a coefficient of a particular power of s . Each DDD seen from a root is called a *coefficient* DDD, and the resulting multi-rooted DDD is called an *s-expanded* DDD. The original DDD is referred to as the *complex* DDD as complex frequency variable s appears in some vertices throughout the rest of the chapter. Such a representation exploits the sharing among different coefficients in a polynomial in addition to that explored by complex DDDs. In Fig. 9, 18 non-terminal vertices are used. In comparison, without exploiting the sharing and the sparsity, $108 (= 12 \times 9, \#product\text{-terms} \times \#symbols)$ vertices would be used.

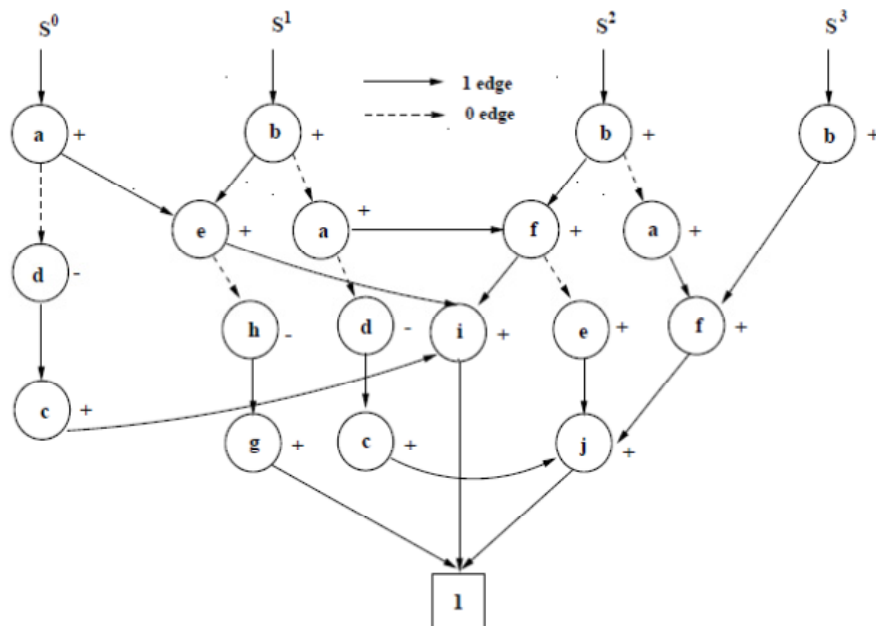


Figure 9: An s-expanded DDD by the first labeling scheme. (Copyright © 2001, Re-printed with permission from [12])

Note that each vertex in a complex DDD may be mapped into several vertices, $a_i, i = 1, \dots, m$, in the resulting s-expanded DDD. We say that a contains a_i and denote this relationship by $a_i \subset a$. As a result, a product term, p , in a complex DDD will generate

a number of product terms, p_i , $i = 1, \dots, l$, in the resulting s-expanded DDD. Similarly, we say p contains p_i and denotes this relationship by $p_i \subset p$. If we further define the row and the column indices of a vertex a_i in a coefficient DDD as that of a , $a_i \subset a$, respectively, we have the following result:

Theorem 3 *A coefficient DDD represents the sum of all the s-expanded product terms of particular power of s in the s-expanded polynomial of a determinant.*

Theorem 3 implies that an s-expanded DDD shares the same properties as a complex DDD, although it does not represent a determinant, instead only those terms that have the same powers of s in a determinant. All the manipulations of complex DDDs mentioned in the section 2 therefore can be applied to s-expanded DDDs.

Under a fixed vertex ordering of all vertices representing admittance parameters in a circuit matrix, the representation of the circuit-matrix determinant by an s-expanded DDD is also canonical. The canonical property in an s-expanded DDD ensures that the maximum sharing among all its coefficients is attained, and the size of the resulting s-expanded DDD is a minimum under a vertex ordering.

If we adopt the second labeling scheme, the same three product terms in the complex DDD of the example circuit will be expanded into 23 product terms in different powers of s :

$$(a + b + cs)(f + g + hs)(k + ls) \rightarrow \begin{cases} +afks^0 & +bgls^1 & +agks^0 & +a/ks^1 \\ +bfks^0 & +b/ks^1 & +bgks^0 & +a/ks^2 \\ +cfks^1 & +b/ls^2 & +cgks^1 & +c/ks^2 \\ +afls^1 & +cfls^2 & +agls^1 & +cgl^2 \\ +afks^1 & +c/ls^3 & & \end{cases}$$

$$(a + b + cs)(-j)(i) \rightarrow \begin{cases} -ajis^0 \\ -bjis^0 \\ -cjis^1 \end{cases},$$

$$(-e)(d)(k + ls) \rightarrow \begin{cases} -edks^0 \\ -edls^1 \end{cases}.$$

The resulting s-expanded DDD is depicted in **Fig. 10**. It is easy to see that the second labeling scheme results in more vertices than the first one. The resulting s-expanded DDD has the same properties as the previous one (using the first labeling scheme), but it will be more suited for the DDD-based approximation to be presented in section 6.

5.2. Construction of s-expanded DDDs

An s-expanded DDD can be constructed from a complex DDD by one depth- first search of the complex DDD. The procedure is very efficient with the time complexity linear in the size of the resulting s-expanded DDD.

For convenience, we first present the construction algorithm using the first labeling scheme. Let D be a complex DDD vertex, with its 1-edge pointing to D_1 and its 0-edge pointing to D_0 . Let $D.g$, $D.c$ and $D.l$ denote, respectively, the admittance of the

conductance, capacitance and inductance in the circuit. An s -expanded DDD, P , is a list of coefficient DDDs with $P[i]$ denoting the coefficient DDD of power s^i and $i \in [-n, n]$. Then, we introduce the following four basic operations:

- $\text{COEFFUNION}(P_1, P_2)$ computes the UNION of two s -expanded DDDs, P_1 and P_2 .
- $\text{COEFFMULPLTY}(P, D.x)$ computes the product of s -expanded DDD P and coefficient in DDD vertex $D.x$.
- $P * s$ increments the power of s in s -expanded DDD P .
- P / s decrements the powers of s in s -expanded DDD P .

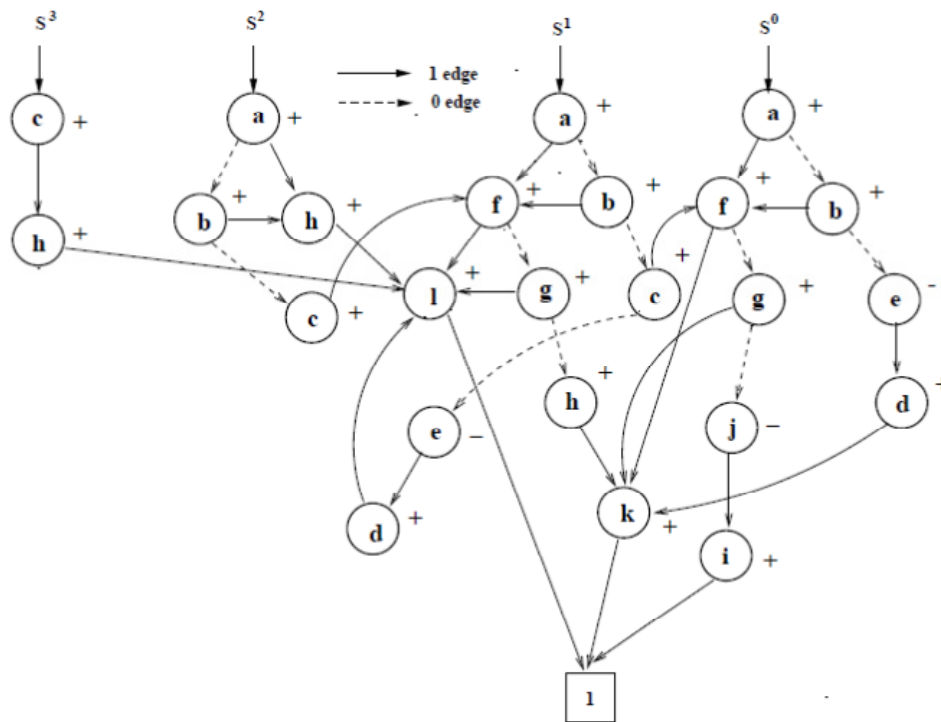


Figure 10: An s -expanded DDD by the second labeling scheme. (Copyright © 2004, Re-printed with permission from [21])

Algorithm COEFFCONST described in **Fig. 11** takes a complex DDD vertex and creates its corresponding coefficient DDDs. The implementations of COEFFUNION and COEFFMULPLTY are also shown in **Fig. 12** in terms of the basic DDD operations MULTIPLY and UNION , whose implementations can be found in **Fig. 4** in section 3.

As in all other DDD operations [1], we cache the result of $\text{COEFFCONST}(D)$, and in case D is encountered again, and its result will be used directly. In the second labeling scheme, we use $D.x_i$ to represent the i th admittance parameter in a complex DDD vertex D . $D.x_i$ can be a resistive admittance, a capacitive admittance or an inductive admittance. The function $\text{type}(D.x_i)$ will return *res*, *cap* and *ind* for the three admittance types, respectively. The COEFFCONST using the second labeling scheme is expressed in **Fig. 13**.

Consider a $n \times n$ circuit matrix. A complex DDD D_r with its size denoted by $|D_r|$ is used to represent the determinant of the circuit matrix. Let n be the size of the determinant D_r represents. The maximum number of the circuit admittance parameters in an entry of a circuit matrix is k . Then, we have the following result for the s -expanded DDD derived from D_r by COEFFCONST for both labeling schemes [12]:

```

COEFFCONST( $D$ )
1  if ( $D = 0$  or  $D = 1$ )
2  return NULL
3   $L_0 = \text{COEFFCONST}(D_0)$ 
4   $L_1 = \text{COEFFCONST}(D_1)$ 
5  if ( $D.g \neq 0$ )
6     $P_g = \text{COEFFMULPLTY}(L_1, D.g)$ 
7  if ( $D.c \neq 0$ )
8     $P_c = \text{COEFFMULPLTY}(L_1 * s, D.c)$ 
9     $P_{\text{result}} = \text{COEFFUNION}(P_c, P_g)$ 
10 if ( $D.l \neq 0$ )
11    $P_l = \text{COEFFMULPLTY}(L_1/s, D.l)$ 
12    $P_{\text{result}} = \text{COEFFUNION}(P_l, P_{\text{result}})$ 
13 return  $\text{COEFFUNION}(P_{\text{result}}, L_0)$ 

```

Figure 11: The s -expanded DDD construction with the first labeling scheme.

Theorem 4 *The time complexity of $\text{COEFFCONST}(D_r)$ and the number of vertices (size) of the resulting s -expanded DDD are $O(kn|D_r|)$.*

Proof. Function $\text{COEFFCONST}(D_r)$ performs a depth-first search on D_r , so it will visit each DDD vertex just once, and COEFFCONST will be called just $|D_r|$ times.

```

COEFFUNION( $P_1, P_2$ )
1  for  $i = -n$  to  $n$  do
2     $P[i] = \text{UNION}(P_1[i], P_2[i])$ 
3  return  $P$ 

COEFFMULPLTY( $P, D.x$ )
1  for  $i = -n$  to  $n$  do
2     $P[i] = \text{MULTIPLY}(P[i], D.x)$ 
3  return  $P$ 

```

Figure 12: The basic DDD construction algorithms


```

COEFFCONST( $D$ )
1  if ( $D = 0$  or  $D = 1$ )
2  return NULL
3   $L_0 = \text{COEFFCONST}(D_0)$ 
4   $L_1 = \text{COEFFCONST}(D_1)$ 
5   $P_{result} = \text{NULL}$ 
6  for  $i = 1$  to  $k$  do
7    if ( $\text{type}(D.x_i) \neq \text{res}$ )
8       $P_g = \text{COEFFMULPLTY}(L_1, D.x_i)$ 
9       $P_{result} = \text{COEFFUNION}(P_g, P_{result})$ 
10   if ( $\text{type}(D.x_i) \neq \text{cap}$ )
11      $P_c = \text{COEFFMULPLTY}(L_1 * s, D.x_i)$ 
12      $P_{result} = \text{COEFFUNION}(P_c, P_{result})$ 
13   if ( $\text{type}(D.x_i) \neq \text{ind}$ )
14      $P_l = \text{COEFFMULPLTY}(L_1/s, D.x_i)$ 
15      $P_{result} = \text{COEFFUNION}(P_l, P_{result})$ 
16  return  $\text{COEFFUNION}(P_{result}, L_0)$ 

```

Figure 13: The s -expanded DDD construction with the second labeling scheme.

6. DDD-Based Symbolic Approximation

Deriving interpretable symbolic small-signal characteristics of analog integrated circuits by approximation can build the circuit behavioral models and gain intuitive insights into the circuit behavior. In this section, we present one efficient algorithm for obtaining approximate symbolic expressions based on DDD representation of symbolic expressions. We show that a dominant term of a determinant can be found by searching shortest paths in the DDD graphs in a linear time in terms of the DDD graph size. Finding the k dominant product terms can be obtained by an incremental k shortest path search algorithm.

Before we generate the dominant terms, one problem we need to consider is symbolic cancellation. Symbolic canceling terms arise from the use of the MNA formulation in analog circuits. For instance, consider the s -expanded DDD shown in **Fig. 9**. Since $g = k = 1/R_3$ and $i = j = -1/R_3$, term $agks^0$ cancels term $-ajis^0$ in the coefficient DDD of s^0 . Our experiments show that 70-90% terms are canceling terms. Clearly it is inefficient to generate the 70%-90% terms that will not show up in the final expressions de-cancellation. It will be shown in [13] that fundamentally symbolic cancellation is caused by the sub-matrix reduction or variable/node reduction. MNA formulation is obtained by reducing all the branch current and branch voltage variables from the sparse tableau formulation, which is cancellation-free [14]. Such a reduction will lead to the symbolic cancellation [15]. More detailed treatment of this issue is covered in [6, 11].

It turns out that symbolic canceling terms can be efficiently removed during [6] or after the s-expanded DDD construction [12]. In the following, we assume we start with the cancellation-free DDDs.

6.1. Finding dominant terms by incremental k-shortest path algorithm

In the following, we present an efficient algorithm for finding k dominant terms. The algorithm does not require DDDs to satisfy certain graph theoretical property required by the dynamic programming based method [17, 18] and thus can be applicable to any DDD graph.

The shortest path (SP) algorithm is based on the observation that the most significant term in coefficient DDDs can be transformed into the SP in edge-weighted DDD graphs by introducing the following edge weight in a DDD:

- 0 -edge costs 0
- 1 -edge costs $-\log|a_i|$, and $|a_i|$ denotes the numerical value of the DDD vertex a_i that originates the corresponding 1 -edge.

The weight of a path in a coefficient DDD is defined to be the total weights of the edges along the path from the root to the 1 -terminal. As a result, given a path, say $abcdef$, their path weight is

$$- (\log |a| + \log |b| + \log |c| + \log |d| + \log |e| + \log |f|). \quad (7)$$

If $|abcdef|$ is the value of the largest term, the value of $-\log|abcdef|$ will be the smallest, which actually is (7).

The shortest (weighted) path in a coefficient DDD, which is a DAG (direct acyclic graph), can be found by depth-first search in time $O(V + E)$, where V is the number of DDD vertices and E is number of edges [19]. So it is $O(V)$ in DDDs. Once we find the shortest path from a DDD, we can subtract it from the DDD using SUBTRACT() operation [1], and then we can find the next shortest path in the resulting DDD.

We can further speed up the process of finding the dominant paths. After every vertex has been visited once (i.e. after the first dominant term is found), the new algorithm is based on the observation that not all the vertices are needed to be visited, after the DDD graph is modified due to the subtraction of a dominant term from the graph. The new algorithm avoid applying the shortest path search algorithm to the DDD graph directly every time, which requires visiting every vertex in a DDD graph to find the dominant term as required by the shortest path search algorithm [19]. We show that only the newly added DDD vertices are needed to be relaxed and the number of newly added DDD vertices is bounded by the depth of a DDD graph.

In the sequel, we first introduce the concept of reverse DDD graphs. As shown in **Fig. 10**, a DDD graph is a direct graph with two terminal vertices and one root vertex. Remember that the 1 -path in a DDD graph is defined from the root vertex to the 1 -terminal. Now we define a new type of DDD graphs, called reverse DDD graphs, where all the edges have their directions reversed and the root of the new graph are 1 -terminal and 0 -terminal vertices and new terminal vertex becomes the root vertex of

the original DDD graph. The reverse DDD graph for the DDD graph in Fig. 8 is shown in Fig. 14. For the clarification, the root vertex and terminal vertices are still referred to as those in the original DDD graphs.

With the concept of the reverse DDD graph, we further define 1-path and path weight in a reverse DDD graph.

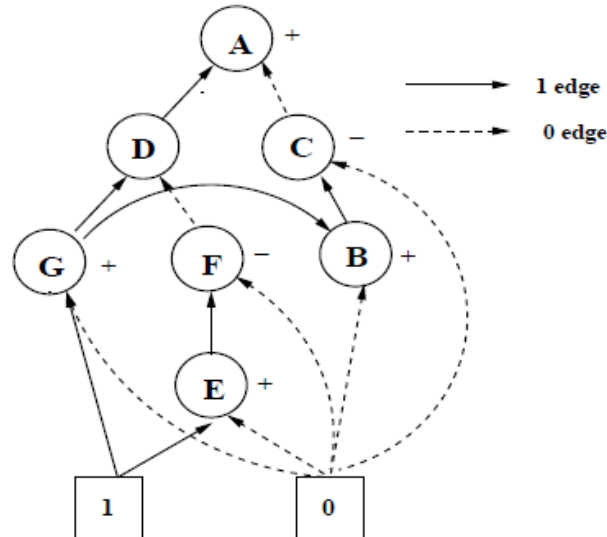


Figure 14: A reverse DDD. (Copyright © 2004, Re-printed with permission from [21])

Definition 1 A 1-path in a reverse DDD is defined as a path from the 1-terminal to root vertex (A in our example) including all symbolic symbols and signs of the vertices that the 1-edges point to along the 1-path.

Definition 2 The weight of a path in a DDD is defined to be the total weights of the edges along the path where each 0-edge costs 0 and each 1-edge costs $-\log|a_i|$, and $|a_i|$ denotes the numerical value of the DDD vertex a_i that the corresponding 1-edge points to.

We then have the following result.

Lemma 1 The most significant product (dominant) term in a symbolic determinant D corresponds to the minimum cost (shortest) path in the corresponding reverse DDD between the 1-terminal and the root vertex.

The shortest path in a reverse s-expanded DDD, which is still a DAG and thus, can be found in $O(|\text{DDD}|)$ time as the normal DDD graph does.

Following the same strategy in [16], after we find the shortest path from a DDD, we can subtract it from the DDD using SUBTRACT() DDD operation, and then we can find the next shortest path in the resulting DDD. We have the following result:

Lemma 2 In a reverse DDD graph, after all the vertices have been visited (after finding the first shortest path), the next shortest path can be found by only visiting newly added vertices created by the Subtraction operation.

Fig. 15 illustrates the incremental k -shortest path. The figure in the left-hand side shows the consecutive k -shortest path algorithm to find the shortest path. Every time when a new DDD graph is created which is rooted at D' , we have to visit the whole graph to find the shortest path. The figure shown in the right-hand side is the new incremental k -shortest path algorithm where we only need to visit all the newly created DDD nodes (in the upper left triangle) to be able to find the shortest path. As shortest paths are found from the source to all the nodes in a graph, the shortest paths, shown in dashed lines, in the existing sub-graphs can be reused in the new DDD graph.

It turns out that finding the shortest path from 1-terminal to the new vertices can be done very efficiently when those new vertices get created. The shortest path searching can virtually take no time during the subtraction operation. Suppose that every vertex in reverse DDD graph D has a shortest path from 1-terminal to it (be visited once). Then the new algorithm for searching the next dominant term is given in **Fig. 16**.

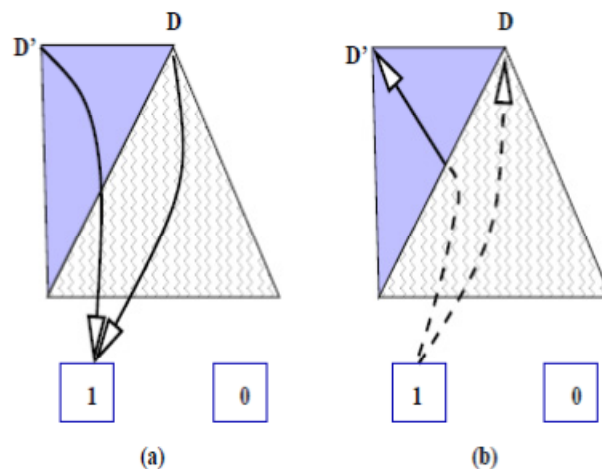


Figure 15: Incremental k -shortest path algorithm. (Copyright © 2004, Re-printed with permission from [21])

In $\text{GETNEXTSHORTESTPATH}(D)$, $\text{EXTRAPATH}(D)$ obtains the found shortest path from D and returns the path in a single DDD graph form. This is done by simply traversing from the root vertex to 1-terminal. Each vertex will remember its immediate parent who is on the shortest path to the vertex in a fully relaxed graph (relaxation concept will be explained soon). Once the shortest path is found, we *subtract* it from the existing DDD graph and relax the newly created DDD vertices (line 15-17) at same time to find the shortest paths from 1-terminal to those vertices, which is performed in the modified function $\text{SUBTRACT}(D,P)$, now called $\text{SUBTRACTANDRELAX}(D, P)$.

In function $\text{SUBTRACTANDRELAX}(D, P)$, $\text{RELAX}(P, Q)$ performs the relaxation operation, an operation that checks if a path from a vertex's parent is the shortest path seen so far and remember the parent if it is, for vertices P and Q where P is the immediate parent of Q in the reverse DDD graph. The ration relaxation is shown in **Fig. 17**. Here, $d(x)$ is the shortest path value seen so far or vertex x ; $w(P,Q)$ is the weight of the edge from P to Q , which actually is the circuit parameter value that Q represents in the reverse DDD graph.

```

GetNextShortestPath(D)
1  if (D = 0)
2      return 0
3  P = EXTRAPATH(D)
4  if (P exists and P not equal to 1)
5      D = SUBTRACTANDRELAX(D, P)
6  return P

SUBTRACTANDRELAX(D, P)
01 if (D = 0)
02     return 0
03 if (P = 0)
04     return D
05 if (D = P)
06     return 0
07 if (D.top > P.top)
08     V = GETVERTEX(D.top, D.child1,
                     SUBTRACTANDRELAX(D.child0, P))
09 if (D.top < P.top)
10     V = SUBTRACTANDRELAX(D, P.child0)
11 if (D.top = P.top)
12     T1 = SUBTRACTANDRELAX(D.child1, P.child1)
13     T0 = SUBTRACTANDRELAX(D.child0, P.child0)
14     V = GETVERTEX(D.top, T1, T0)
15 if (V not equal to D)
16     RELAX(V.child1, V)
17     RELAX(V.child0, V)
18 return V

```

Figure 16: Incremental k-shortest path based dominant term generation algorithm. (Copyright © 2004, Re-printed with permission from [21])

```

RELAX(P, Q)
1  if  $d(Q) > d(P) + w(P, Q)$ 
2      $d(Q) = d(P) + w(P, Q)$ 
3     parent(Q) = P

```

Figure 17: The RELAX() operation. (Copyright © 2004, Re-printed with permission from [21])

Line $parent(Q) = P$ remembers the parent of Q in the shortest path from the 1-terminal to Q . In the reverse DDD graph, each vertex has only two incoming edges (from its two children in the normal DDD graph), so the relaxation with its two parents in lines 16 and 17 are sufficient for the relaxation of vertex V . Moreover, the relaxation for V happens after all its parents have been RELAXed due to the DFS-type traversal in SUBTRACTANDRELAX(). This is consistent with the ordering requirement of the shortest path search algorithm. Therefore by repeatedly invoking function GETNEXTSHORTESTPATH(D), we can find all the dominant terms in a decreasing order.

Let n be the number of vertices in a path from 1-terminal to the root vertex, i.e. the depth of the DDD graph, given the fact that D is a DDD graph and P is a path in the DDD form, then we have the following theorem:

Theorem 5 *The number of new DDD vertices created in function SUBTRACTANDRELAX(D, P) is bounded by n and the time complexity of the function is $O(n)$.*

We then have the following result for incremental k-SP based algorithm:

Theorem 6 *The time complexity of the incremental k-SP algorithm for finding k shortest paths is*

$$O(|DDD| + n(k - 1)), \quad (2.9)$$

where n is the depth of the DDD graph.

Notice that both DP based algorithm and incremental k-SP based algorithm have time complexity $O(|DDD|)$ to find a dominant term, where $|DDD|$ is the size of a DDD graph. After the first dominant term, however, both algorithms show better time complexities for generating next dominant terms, that is $O(n)$. But in contrast to DP based algorithm, the actual running time of the incremental k-SP based algorithm does not depend on the topology of a circuit.

Notice that the new incremental k-shortest path generation algorithm can be performed on any DDD graph, including cancellation-free s -expanded DDD. We note that the variant of DDD used by Verhaegen and Gielen in [17, 18] does not satisfy the canonical property due to vertex duplication. As a result, except for the first shortest path, remaining dominant paths cannot easily be generated by using the shortest path algorithm as the found shortest path is hard to be SUBTRACTED (if possible at all) as most DDD graph operations are not valid for a non-canonical DDD graph.

Following the same strategy in [15], our approach also handles numerical cancellation. Since numerical canceling terms are extracted one after another, they can be eliminated by examining two consecutive terms.

7. Conclusion

In this chapter, we briefly review the determinant decision diagram (DDD) concepts and its application for symbolic analysis and generating the dominant symbolic terms for analog behavioral modeling. We start with the basic concept of a DDD, its main

properties and manipulative operations for symbolic analysis. Then we introduce an efficient DDD construction algorithm by logic synthesis and operation. We then present s-expanded DDDs to represent s-expanded polynomials and s-domain transfer functions. Finally we give a shortest-path- searching based algorithm for finding k dominant symbolic terms for symbolic approximations. The new algorithm has a linear time complexity in terms of DDD graphs and can find k dominant terms very efficiently.

This chapter covers all the basic essence about DDD-based symbolic analysis. We do not include many proof details and numerical results. Interesting readers can refer to more detailed treatment of DDD graphs and application in [6, 11].

Acknowledgements

The author acknowledges the support from UC-MEXUS-CONACYT collaboration grant and the support from NSF grant under No. CCF-0448534.

References

- [1] C.-J. Shi and X.-D. Tan, "Canonical symbolic analysis of large analog circuits with determinant decision diagrams," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, vol. 19, Jan. 2000, pp. 1–18.
- [2] R. E. Bryant, "Graph-based algorithms for Boolean function manipulation," *IEEE Transactions on Computers*, 1986, pp. 677–691.
- [3] S. Minato, "Zero-suppressed BDDs for set manipulation in combinatorial problems," in *Proc. Design Automation Conference*, 1993, pp. 272–277.
- [4] S. B. Akers, "Binary decision diagrams," *IEEE Transactions on Computers*, vol. 27, no. 6, 1976, pp. 509–516.
- [5] G. D. Hachtel and F. Somenzi, *Logic synthesis and verification algorithm*. Kluwer Academic Publishers, 1996.
- [6] Z. Qin, S. X.-D. Tan, and C. Cheng, *Symbolic analysis and reduction of VLSI circuits*. Boston, MA: Kluwer Academic Publishers, 2005.
- [7] R. L. R. K. S. Brace and R. E. Bryant, "Efficient implementation of a BDD package," in *Proc. Design Automation Conference*, 1990, pp. 40–45.
- [8] P. M. Lin, *Symbolic network analysis*. Elsevier Science Publishers B.V., 1991.
- [9] J. Zhu and S. Calman, "Symbolic pointer analysis revisited," in *Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation*, June 2004.
- [8] G. H. Golub and C. F. V. Loan, *Matrix computations*. Baltimore, MD: The Johns Hopkins University Press, 3 ed., 1989.
- [11] S. X.-D. Tan, *Symbolic analysis of large analog circuits with determinant decision diagrams*. University of Iowa: Ph.D. Thesis, 1999.
- [12] C.-J. Shi and X.-D. Tan, "Compact representation and efficient generation of s-expanded symbolic network functions for computer-aided analog circuit design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 20, April 2001, pp. 813–827.
- [13] X.-D. Tan and C.-J. Shi, "Hierarchical symbolic analysis of large analog circuits via determinant decision diagrams," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, vol. 19, April 2000, pp. 401–412.
- [14] J. Vlach and K. Singhal, *Computer methods for circuit analysis and design*. New York, NY: Van Nostrand Reinhold, 1995.
- [15] S. X.-D. Tan, "A general s-domain hierarchical network reduction algorithm," in *Proc. Int. Conf. on Computer Aided Design*, Nov. 2003, pp. 650–657.
- [16] X.-D. Tan and C.-J. Shi, "Interpretable symbolic small-signal characterization of large analog circuits using determinant decision diagrams," in *Proc. European Design and Test Conference (DATE)*, 1999, pp. 448–453.
- [17] W. Verhaegen and G. Gielen, "Efficient ddd-based symbolic analysis of large linear analog circuits," in *Proc. Design Automation Conference*, June 2001, pp. 139–144.

- [18] W. Verhaegen and G. Gielen, "Symbolic determinant decision diagrams and their use for symbolic modeling of linear analog integrated circuits," *Analog Integrated Circuits and Signal Processing*, vol. 31, May 2002, pp. 119–130.
- [19] T. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to algorithms*. Cambridge, Massachusetts: The MIT Press, 1990.
- [20] S. X.-D. Tan and C.-J. Shi, "Efficient approximation of symbolic expressions for analog behavioral modeling and analysis." *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, vol. 23, no.6, pp. 907-918, June 2004.
- [21] S. X-D. Tan, "Symbolic analysis of analog circuits by Boolean logic operations", *IEEE Transactions on Circuits and Systems II*, vol. 53, no. 11, pp.1313-1317, Nov. 2006.

Sensitivity Computation Based on Auxiliary Circuits

Lucia Dumitriu* and Mihai Iordache

Electrical Engineering Department, Politehnica University of Bucharest, Romania

Abstract: Some aspects deriving from the circuit element manufacturing, the environmental conditions (temperature, humidity, radiation) and also the normal aging, can alter the circuit operation performance by changing the element parameter values. That is why the design of such systems has to take into account the effect of the element parameter variations on the circuit performance. A precise measure of this effect is done by the sensitivity function that can be performed for the magnitude of the transfer function, for the natural frequencies of the circuit, for the quality factor *etc.* Knowledge of the circuit sensitivity can be used as a basis for comparing different electronic circuits. It helps the circuit designer in selecting the proper circuit for a specified application. The chapter presents the most popular definitions of the sensitivity and develops three methods to compute these important elements for CAD of electric circuits. Illustrative examples are presented and important remarks are done.

Keywords: Symbolic analysis, circuit analysis, computer aided design, sensitivity, sensitivity analysis, sensitivity computation, auxiliary circuit, incremental circuit, adjoint circuit, Bykhovski method.

1. Introduction

Consider a linear circuit having the performance denoted by \mathcal{P} , and the parameter vector $\mathbf{x} = \{x_1, \dots, x_k, \dots, x_n\}$, $\mathcal{P} = \mathcal{P}(\mathbf{x})$. At a small variation δx_k of the parameter x_k , the Taylor series of $\delta \mathcal{P}$ around the nominal value x_{k0} is:

$$\delta \mathcal{P}(\mathbf{x}) = \mathcal{P}(x_k) - \mathcal{P}(x_{k0}) = \frac{\partial \mathcal{P}}{\partial x_k} \delta x_k + \frac{1}{2} \frac{\partial^2 \mathcal{P}}{\partial x_k^2} \delta^2 x_k + \dots \quad (1)$$

Definition 1.

The most used definition of the circuit performance sensitivity, evaluated as the circuit transfer function ($\mathcal{P} = H$), with respect to the parameter x_k , at small changes in the parameter value, denoted by $S_{x_k}^H$, is defined as:

$$S(H, x_k) = S_{x_k}^H \stackrel{d}{=} \lim_{\delta x_k \rightarrow 0} \frac{\delta H}{\delta x_k} = \frac{\partial H}{\partial x_k} \quad (2)$$

Definition 2.

The relative sensitivity (or normalized sensitivity) definition of the transfer function H , with respect to the parameter x_k , at small changes in the parameter value, is

$$S(H, x_k) = S_{x_k}^H \stackrel{d}{=} \lim_{\delta x_k \rightarrow 0} \frac{\delta H / H}{\delta x_k / x_k} = \frac{x_k}{H} \frac{\partial H}{\partial x_k} = \frac{\partial \ln H}{\partial \ln x_k} \quad (3)$$

When the excitations are sinusoidal signals, the circuit transfer function $H(j\omega)$ can be expressed as:

*Address correspondence to Lucia Dumitriu: Electrical Engineering Department, Politehnica University of Bucharest, Romania; E-mail: lucia.dumitriu@gmail.com

$$H(j\omega) = |H(j\omega)|e^{j\varphi} \quad \text{or} \quad \ln H(j\omega) = \ln|H(j\omega)| + j\varphi = A(\omega) + j\varphi \quad (4)$$

where: $A(\omega) = \ln|H(j\omega)|$ is the magnitude, and $\varphi = \arg(H(j\omega))$ is the phase of the signal.

The sensitivities of the magnitude and of the phase, respectively, in respect of the parameter x_k are defined as:

$$S(A(\omega), x_k) = x_k \frac{\partial A(\omega)}{\partial x_k} \quad (5)$$

and

$$S(\varphi(\omega), x_k) = x_k \frac{\partial \varphi(\omega)}{\partial x_k} \quad (6)$$

The relative sensitivity of the transfer function (4) in respect of the parameter x_k can be written as:

$$S(H(j\omega), x_k) = \frac{\partial(\ln H(j\omega))}{\partial(\ln x_k)} = \frac{\partial(A(\omega) + j\varphi(\omega))}{\partial x_k / x_k} = \frac{\partial A(\omega)}{\partial x_k / x_k} + j \frac{\partial \varphi(\omega)}{\partial x_k / x_k} \quad (7)$$

By computing the sensitivity of the circuit performance in respect of any circuit parameter, the critical circuit elements will be found. These are the elements whose small variations in the parameter values cause important changes in the circuit performance. The circuit sensitivities in respect of these parameters are very high. Identification of the critical/noncritical circuit elements allows building a simplified circuit model that increases the analysis efficiency. In the reduced model the noncritical elements will be removed fulfilling an imposed error criterion for the transfer function [1].

An optimal design, made with efficient tools, has to include a sensitivity analysis that gives to the designer the opportunity to select between two circuits that with smaller sensitivities (meaning smaller costs) while keeping the circuit performance in the desired range.

The unavoidable dispersion of the parameter values in the industrial execution implies knowing the circuit performance in a given range of values, called the tolerance domain. The tolerance defines a domain of the possible values around the nominal value that a parameter can take, without significant alteration of the circuit performance. The sensitivity study facilitates developing the tolerance analysis. The objective of the circuit design is to produce a circuit with reasonable sensitivities and tolerances. It's important to remark that small sensitivities allow a relaxation of the tolerance constraints, resulting in cost reduction.

The above sensitivities are defined in respect of a single parameter. Actually, many parameters change simultaneously, so that the first order approximation of the transfer function relative change is:

$$\frac{\delta H}{H} = \sum_{k=1}^n S(H, x_k) \frac{\delta x_k}{x_k} = \sum_{k=1}^n \frac{\partial(\ln H)}{\partial(\ln x_k)} \delta(\ln x_k) \quad (8)$$

If the incremental changes $\delta(\ln x_k)$ of the circuit parameters are expressed by the vector:

$$\delta \mathbf{y} = [\delta(\ln x_1), \delta(\ln x_2), \dots, \delta(\ln x_n)]^t \quad (9)$$

and the set of gradients $\delta(\ln H)/\delta(\ln x_k)$ by the vector,

$$\nabla \mathbf{G} = \left[\frac{\delta(\ln H)}{\delta(\ln x_1)}, \frac{\delta(\ln H)}{\delta(\ln x_2)}, \dots, \frac{\delta(\ln H)}{\delta(\ln x_n)} \right]^t \quad (10)$$

$\delta H/H$ can be expressed as:

$$\frac{\delta H}{H} = \nabla \mathbf{G}^t \delta \mathbf{y} \quad (11)$$

In this way, the multiparameter sensitivity can be defined [2, 3] as

$$\mathbf{S}_{mp} = \nabla \mathbf{G} \quad (12)$$

When the number of parameters is large, computing the multiparameter sensitivity becomes a very difficult and time consuming task. Therefore, it is very important to apply the appropriate approach for each specified problem.

There are three classes of methods for the computation of multiparameter sensitivity:

1. Methods based on *feedback theory* or the *bilinear theorem* - for linear circuits;
2. *Direct methods*, by which the first-order derivatives of interest (and any desired higher-order derivatives) are computed directly from the circuit matrix (usually nodal admittance matrix);
3. Indirect methods, requiring an *auxiliary circuit* associated with the circuit under consideration.

In the following sections we shall develop three methods for sensitivity calculation, based on auxiliary circuits.

Note. As it results from the transfer function definitions (see Chapter 4, definitions (2)) computing the partial derivatives of a transfer function (sensitivity evaluation) is equivalent to calculation of the voltage or current derivatives [4]. That is why the next proofs will be made in terms of the incremental currents and incremental voltages of the circuit at small changes of the circuit parameters.

2. The Incremental Circuit Applied to Sensitivity Analysis

Considering a part of a linear circuit \mathcal{C} , we wish to evaluate the changes in the branch currents and voltages due to small changes in branch parameters-resistances/conductances (impedances/admittances) and parameters associated to the controlled sources. To this end we shall build a perturbed circuit \mathcal{C}_p , whose branch parameters are changed with slight amounts, while the excitations are unperturbed. The two circuits \mathcal{C} and \mathcal{C}_p have the same topology. For simplicity reasons we shall consider a resistive circuit \mathcal{C}_R (**Fig. 1(a)**).

When the resistance R_k takes the value $R_k + \Delta R_k$, the branch currents and voltages will be also modified. If we consider the incidence reduced matrix \mathbf{A} and the fundamental loop matrix \mathbf{B} , the Kirchhoff's laws for the two circuits take the form:

$$\mathbf{A}\mathbf{I} = \mathbf{0}, \quad \mathbf{B}\mathbf{V} = \mathbf{0} \quad (13)$$

for the original circuit \mathcal{C}_R , and

$$\mathbf{A}(\mathbf{I} + \Delta \mathbf{I}) = \mathbf{0}, \quad \mathbf{B}(\mathbf{V} + \Delta \mathbf{V}) = \mathbf{0} \quad (14)$$

for the perturbed circuit \mathcal{C}_p (Fig. 1(b)).

From (13) and (14) it results:

$$A\Delta I = \mathbf{0}; \quad B\Delta V = \mathbf{0}, \quad (15)$$

meaning that ΔI and ΔV could be the branch currents and voltages of a circuit \mathcal{C}_i called *the incremental circuit*, that has the same topology as \mathcal{C}_R and \mathcal{C}_p but specific branch characteristics, as in Fig. 1(c).

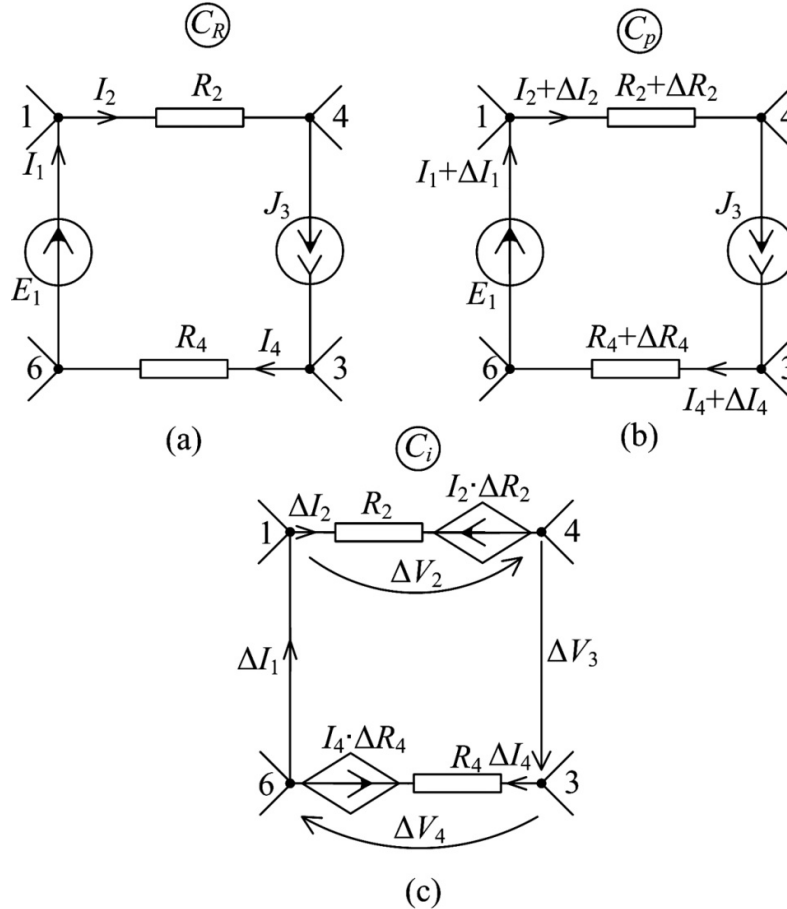


Figure 1: Generation of the incremental circuit.

These characteristics will be identified as follows.

The branch having the resistance R_k from the circuit \mathcal{C}_R , satisfies the equation

$$V_k = R_k I_k \quad (16)$$

In the perturbed circuit \mathcal{C}_p , the same branch satisfies the equation:

$$V_k + \Delta V_k = (R_k + \Delta R_k)(I_k + \Delta I_k) = R_k I_k + R_k \Delta I_k + I_k \Delta R_k + \Delta R_k \Delta I_k \quad (17)$$

Taking into account (16), from (17) it results:

$$\Delta V_k = R_k \Delta I_k + I_k \Delta R_k + \Delta R_k \Delta I_k \quad (18)$$

Because the changes are infinitesimally small, we can neglect the second-order term, so that (18) becomes:

$$\Delta V_k = R_k \Delta I_k + I_k \Delta R_k \quad (19)$$

Equation (19) suggests an equivalent branch consisting in an R_k in series with a current-controlled voltage source $I_k \Delta R_k$ as in **Fig. 1(c)**, and on the first row and the second column in **Fig. 2**. The controlling current I_k belongs to the original circuit \mathcal{C}_R .

Because the independent sources in \mathcal{C}_R are unperturbed, $\Delta \mathbf{J} = \mathbf{0}$ and $\Delta \mathbf{E} = \mathbf{0}$, suggesting that in the incremental circuit \mathcal{C}_i the current source have to be replaced by an open circuit, while the voltage source by a short circuit.

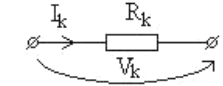
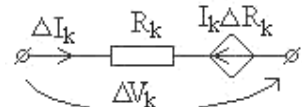
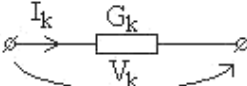
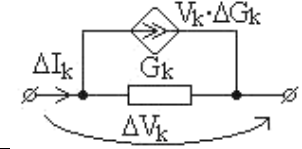
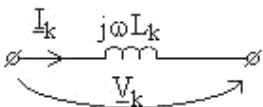
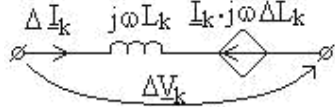
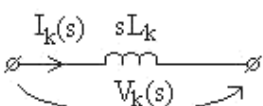
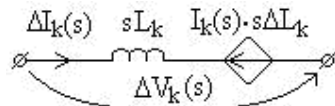
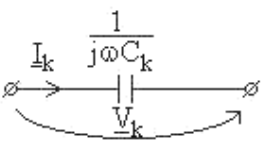
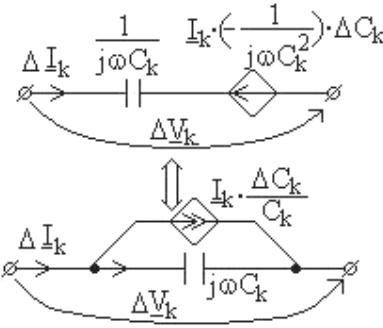
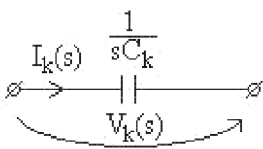
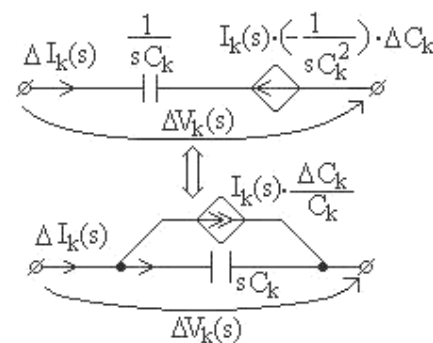
Element in the original circuit \mathcal{C}	Element in the incremental circuit \mathcal{C}_i
	
	
	
	
	
	

Figure 2: cont...

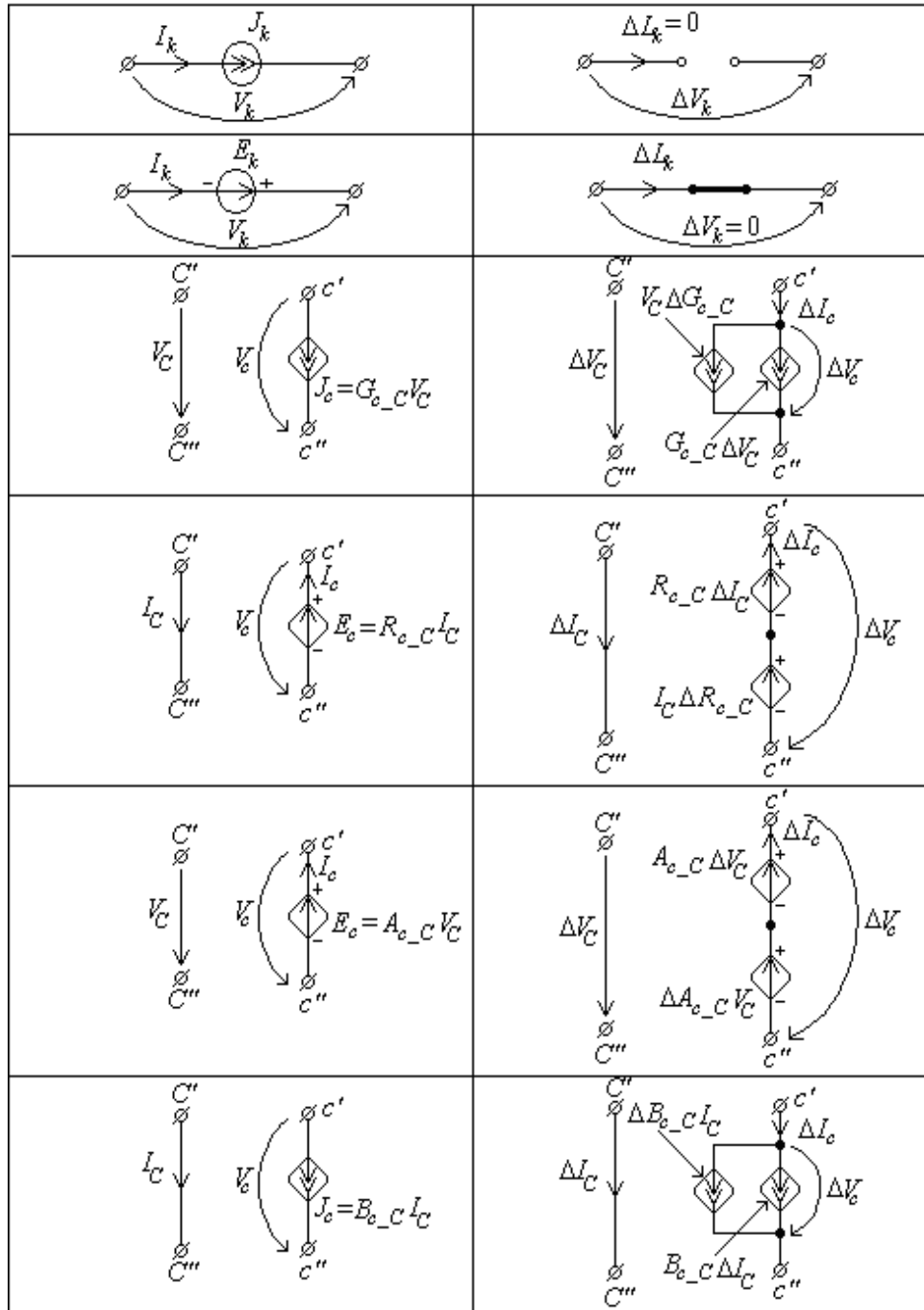


Figure 2: Representation of the elements in the original and in the incremental circuit.

In **Fig. 2** the incremental circuits for the passive circuit elements, independent sources and all the four type of controlled sources are presented.

Usefulness of the incremental circuit for the sensitivity computation will be illustrated later.

Example 1

Consider the nonreciprocal resistive circuit in **Fig. 3(a)**. We want to compute the partial derivatives of the currents I_1, I_2, I_3, I_4 and I_5 in respect of R_1, R_2, R_3, R_5 and

$A_{4,1}$. We want also to compute the sensitivity $\partial G_i / \partial A_{4,1}$. In order to solve the problem, the following algorithm will be applied:

Step 1. The original circuit \mathcal{C}_R is analyzed. The results are given in **Fig. 3(a)**;

Step 2. Taking into account the rules in **Fig. 2**, the incremental circuit is built (**Fig. 3(b)**). Because the controlling currents of the current-controlled voltage sources in series with the resistances R_1 , R_2 , R_3 , and R_5 , (according with **Fig. 2**) are known from the step 1, these sources become independent sources.

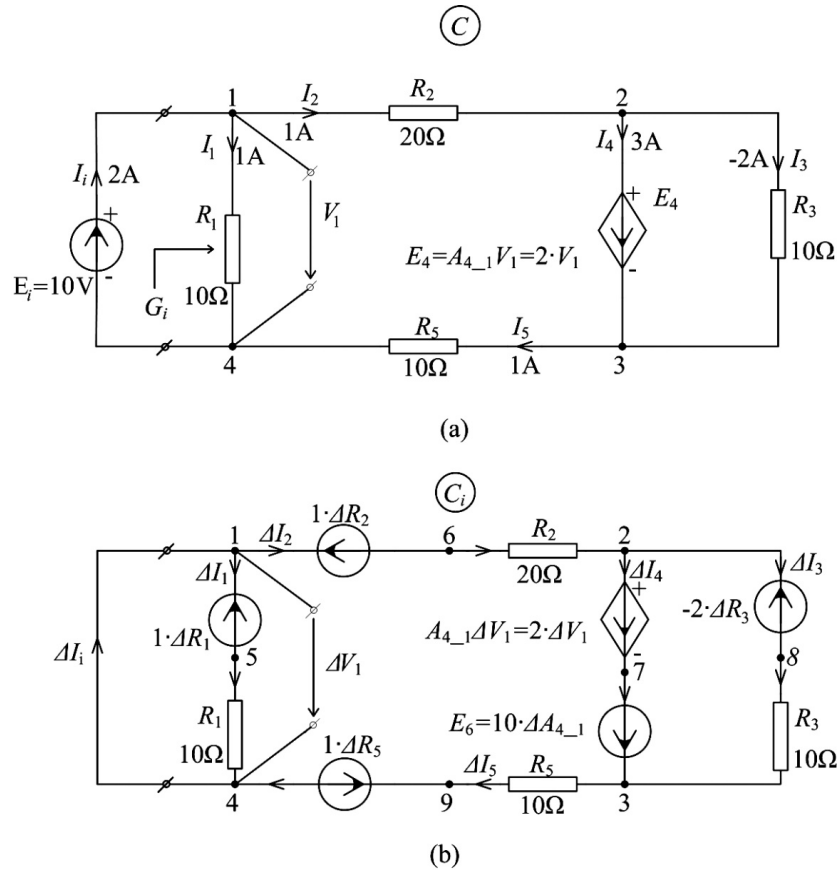


Figure 3: The original circuit (a) and the incremental one (b).

Note. We can see that the two circuits \mathcal{C}_R and \mathcal{C}_i differ only in the location and the values of the independent sources.

Step 3. The incremental circuit \mathcal{C}_i is analyzed by hand and the following results are obtained:

$$\begin{aligned}\Delta I_i &= -\frac{1}{30}\Delta R_2 - \frac{1}{10}\Delta R_3 - \frac{1}{30}\Delta R_5 + \frac{1}{3}\Delta A_{4,1} \\ \Delta I_1 &= -\frac{1}{10}\Delta R_1 \\ \Delta I_2 &= -\frac{1}{30}\Delta R_2 - \frac{1}{30}\Delta R_5 + \frac{1}{3}\Delta A_{4,1} \\ \Delta I_3 &= \frac{1}{5}\Delta R_3 - \Delta A_{4,1}\end{aligned}\quad (20)$$

$$\Delta I_4 = -\frac{1}{30}\Delta R_2 - \frac{1}{5}\Delta R_3 - \frac{1}{30}\Delta R_5 + \frac{4}{3}\Delta A_{4,1}.$$

$$\Delta I_5 = -\frac{1}{30}\Delta R_2 - \frac{1}{30}\Delta R_5 + \frac{1}{3}\Delta A_{4,1}$$

If the SYAMNM program [5] is applied, the above results are found:

```

rez1_ex1_Ci={dE4 = 0., dI1 = -.1000*dR1,
dI2 = -.3334e-1*dR5-.3334e-1*dR2+.3334*dA4_1,
dI3 = -1.000*dA4_1+.2000*dR3, dI5 = -.3333e-1*dR5-.3333e-1*dR2+
+.3333*dA4_1, dI4 = 1.3333*dA4_1-.3333e-1*dR2-.2000*dR3-.3333e-1*dR5,
dIi = dI7 = -.3333e-1*dR5-.3333e-1*dR2+0.3333*dA4_1-.1000*dR3, dU1 = 0.,
dU2 = -.6666*dR5+.3333*dR2+6.666*dA4_1, dU3 = -10.*dA4_1, dU4 = 0.,
dU5 = 3.333*dA4_1+.6666*dR5-.3333*dR2, dU6 = -10.*dA4_1, dU7 = 0.};
    
```

Note. Increasing the analysis efficiency can be obtained by solving the incremental circuit together with the original one. This can be done connecting the two circuits by the reference node as in **Fig. 4**.

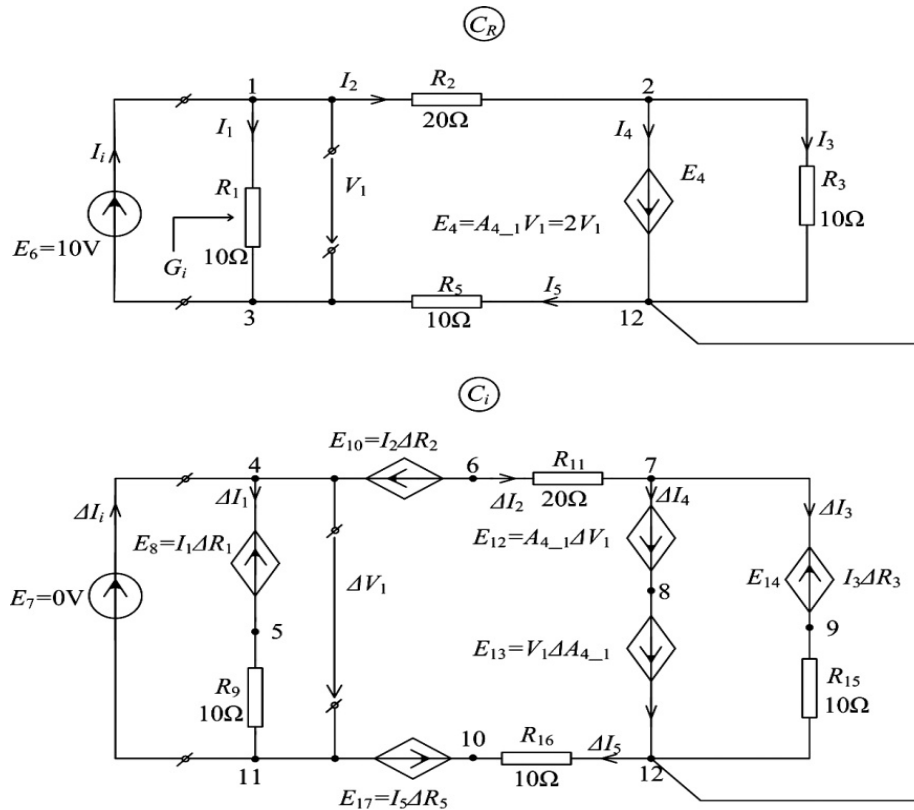


Figure 4: Connection of the original circuit with the incremental one.

In this case the controlling currents of the current-controlled voltage sources from C_i belong to the original circuit.

Using the program, the following results are obtained:

```

rez1_ex1-Complete={dE10 = dR2, dE12 = 0., dE13 = 10.*dA4_1,
dE14 = -2.*dR3, dE17 = dR5, E4 = 20., dE8 = dR1, I1 = 1.,
- dI2 = I10 = .3333e-1*dR5+.3333e-1*dR2- .3333*dA4_1,
    
```


$dI4 = I12 = -.3333e-1*dR5-.3333e-1*dR2+1.333*dA4_1-.2000*dR3,$
 $dI4 = I13 = -.3333e-1*dR5-.3333e-1*dR2+1.333*dA4_1-.2000*dR3,$
 $- dI3 = I14 = 1.000*dA4_1-.2000*dR3, dI3 = I15 = -1.000*dA4_1+.2000*dR3,$
 $dI5 = I16 = -.3333e-1*dR5-.3333e-1*dR2+.3333*dA4_1,$
 $- dI5 = I17 = .3333e-1*dR5+ .3333e-1*dR2-.3333*dA4_1, I2=1., I3= -2., I4= 3., I5 = 1., I6 = 2.,$
 $dIi = I7= -.3333e-1*dR5-.3333e-1*dR2+.3333*dA4_1-.1000*dR3,$
 $- dI1 = I8 = .1000*dR1, dI1 = I9 = -.1000*dR1, U1 = 10., dU10 = -1.*dR2,$
 $dU11 = -.6667*dR5- .6667*dR2+6.667*dA4_1, dU12 = 0.,$
 $dU13 = -10.*dA4_1, dU14 = 2.*dR3, dU15 = -10.00*dA4_1+2.000*dR3,$
 $dU16 = -.3333*dR5-.3333*dR2+3.333*dA4_1, dU17 = -1.*dR5, U2 = 20.,$
 $U3 = -20., U4 = -20., U5 = 10., U6 = -10., dU7 = 0., dU8 = -1.*dR1, dU9 = -1.*dR1\}.$

These results are identical with the above ones.

Step 4. From the above equations we can compute the sensitivities in respect of different parameters at small changes of their values ($\Delta R_1, \Delta R_2, \Delta R_3, \Delta R_5$ and $\Delta A_{4,1} \rightarrow 0$), that actually are the coefficients of the parameters increments:

$$\begin{aligned}
 \frac{\partial I_1}{\partial R_1} &= -\frac{1}{10}, \quad \frac{\partial I_1}{\partial R_2} = 0, \quad \frac{\partial I_1}{\partial R_3} = 0, \quad \frac{\partial I_1}{\partial R_5} = 0, \quad \frac{\partial I_1}{\partial A_{4,1}} = 0 \\
 \frac{\partial I_2}{\partial R_1} &= 0, \quad \frac{\partial I_2}{\partial R_2} = -\frac{1}{30}, \quad \frac{\partial I_2}{\partial R_3} = 0, \quad \frac{\partial I_2}{\partial R_5} = -\frac{1}{30}, \quad \frac{\partial I_2}{\partial A_{4,1}} = \frac{1}{3} \\
 \frac{\partial I_3}{\partial R_1} &= \frac{\partial I_3}{\partial R_2} = \frac{\partial I_3}{\partial R_5} = 0, \quad \frac{\partial I_3}{\partial R_3} = \frac{1}{5}, \quad \frac{\partial I_3}{\partial A_{4,1}} = -1 \\
 \frac{\partial I_4}{\partial R_1} &= 0, \quad \frac{\partial I_4}{\partial R_2} = -\frac{1}{30}, \quad \frac{\partial I_4}{\partial R_3} = -\frac{1}{5}, \quad \frac{\partial I_4}{\partial R_5} = -\frac{1}{30}, \quad \frac{\partial I_4}{\partial A_{4,1}} = \frac{4}{3} \\
 \frac{\partial I_5}{\partial R_1} &= \frac{\partial I_5}{\partial R_3} = 0, \quad \frac{\partial I_5}{\partial R_2} = -\frac{1}{30}, \quad \frac{\partial I_5}{\partial R_5} = -\frac{1}{30}, \quad \frac{\partial I_5}{\partial A_{4,1}} = \frac{1}{3} \\
 \frac{\partial G_i}{\partial A_{4,1}} &= \frac{\partial}{\partial A_{4,1}} \left(\frac{I_i}{E_i} \right) = \frac{1}{E_i} \frac{\partial I_i}{\partial A_{4,1}} = \frac{1}{10} \cdot \frac{1}{3} = \frac{1}{30}
 \end{aligned} \tag{21}$$

Another way to compute the sensitivity $\partial G_i / \partial A_{4,1}$ is by symbolic generation. Performing the simulation we obtain the expression:

$$G_{1,4-1,4} = \frac{R_2 + R_5 + A_{4,1}R_1 + R_1}{R_1(R_2 + R_5)} \tag{22}$$

This expression allows the sensitivity computation based on the definition. The conductance sensitivity in respect to the voltage gain is:

$$S_{A_{4,1}}^{G_{1,4-1,4}} = \frac{\partial G_{1,4-1,4}}{\partial A_{4,1}} = \frac{1}{R_2 + R_5} = \frac{1}{20 + 10} = \frac{1}{30} \tag{23}$$

identical with the last equation from the system (21), obtained by using the incremental circuit.

Example 2

Consider the circuit in **Fig. 5**, and small changes of the parameter values.

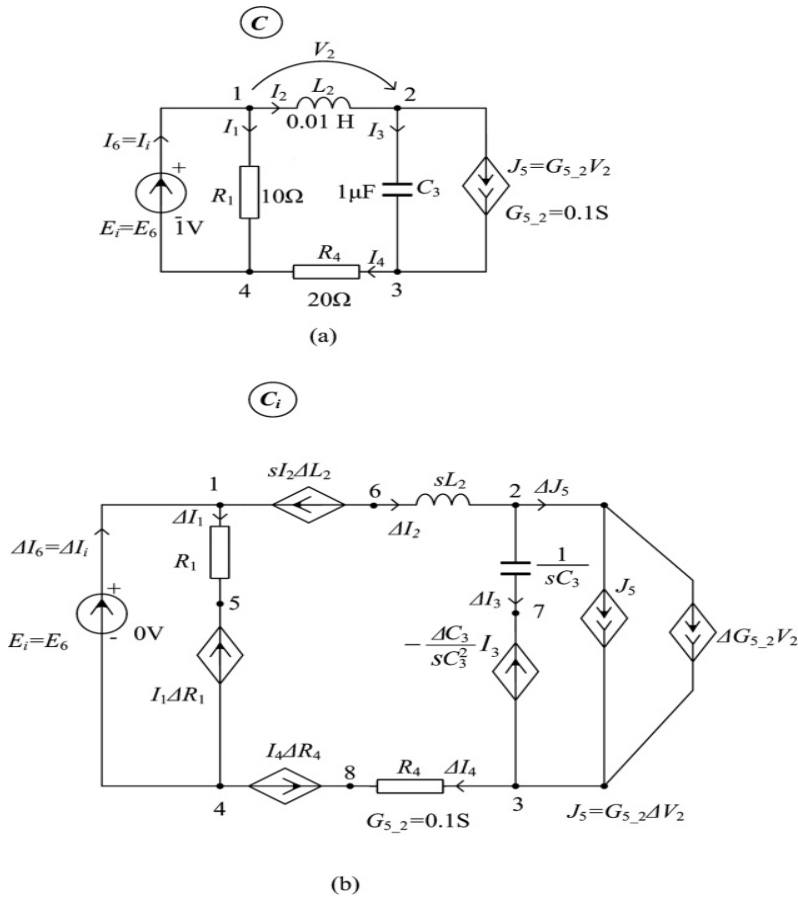


Figure 5: Nonreciprocal RLC circuit: the original circuit (a), and the incremental circuit (b).

The vectors of the branch currents and branch voltages for the original circuit **C** (**Fig. 5(a)**) have the following structure:

$$I_b = [I_1, I_2, I_3, I_4, J_5, I_i]^t, \text{ and } V_b = [V_1, V_2, V_3, V_4, V_5, V_i]^t, \text{ respectively.}$$

Running the SYAMNM program, we get for these circuit variables the values:

$$I_b = \begin{bmatrix} \frac{0.1000}{100. s} \\ \frac{(s - 1031.)(s - 96970.)}{0.1000 s (s - 1000.)} \\ \frac{100. s}{(s - 1031.)(s - 96970.)} \\ \frac{0.1000 s^2}{(s - 1031.)(s - 96970.)} \\ \frac{0.1000 (s - 1042.)(s - 95960.)}{(s - 1031.)(s - 96970.)} \end{bmatrix} \quad V_b = \begin{bmatrix} \frac{1.}{s^2} \\ \frac{(s - 1031.)(s - 96970.)}{100000. s - 0.1000 \cdot 10^9} \\ \frac{2000. s}{(s - 1031.)(s - 96970.)} \\ \frac{100000. s - 0.1000 \cdot 10^9}{(s - 1031.)(s - 96970.)} \\ -1. \end{bmatrix}$$

Using these results, we get for the incremental variations of the branch currents $\Delta I_b = dI_{-i} = [\Delta I_1, \Delta I_2, \Delta I_3, \Delta I_4, \Delta J_5]^t$ in the incremental circuit **C_i** (**Fig. 5(b)**), the following solution:

$$dI_i = \left[\begin{array}{l} \frac{-0.01000R1}{(s-1031)^2(s-96970)^2} \frac{1.5(0.10000l^4s-0.1000l^7)dC3}{(s-1031)^2(s-96970)^2} \frac{1.5(-0.1000l^0s+10000s^2)dL2}{(s-1031)^2(s-96970)^2} \frac{10000s^2dR4}{(s-1031)^2(s-96970)^2} + \frac{0.1000l^0dG5_2}{(s-1031)^2(s-96970)^2} \\ \frac{s(0.1000l^1s^2-0.2000l^4s+0.1000l^7)dC3}{(s-1031)^2(s-96970)^2} \frac{30000s^3dL2}{(s-1031)^2(s-96970)^2} \frac{s(10.00s^2-10000s)dR4}{(s-1031)^2(s-96970)^2} + \frac{s(-1.000s^3-2000s^2)dG5_2}{(s-1031)^2(s-96970)^2} \\ \frac{-1.5(0.1000l^4s-0.1000l^7)dC3}{(s-1031)^2(s-96970)^2} \frac{1.5(-0.1000l^0s+10000s^2)dL2}{(s-1031)^2(s-96970)^2} \frac{10000s^2dR4}{(s-1031)^2(s-96970)^2} + \frac{0.1000l^0dG5_2}{(s-1031)^2(s-96970)^2} \\ \frac{1.5^2(0.1000l^1s-0.1000l^4)dC3}{(s-1031)^2(s-96970)^2} \frac{1.5^2(-20000s-0.1000l^0)dL2}{(s-1031)^2(s-96970)^2} \frac{10.5^3dR4}{(s-1031)^2(s-96970)^2} - \frac{1.5^2(-1.5^2-1999s-0.9998l^0)dG5_2}{(s-1031)^2(s-96970)^2} \end{array} \right]$$

The incremental variation of the input current $dI_i = dI6$ has the expression:

$$dI_i = \frac{1.(0.1000l^4s^2-0.1000l^7s)dC3}{(s-1031)^2(s-96970)^2} \frac{1.(-0.1000l^0s^2+10000s^3)dL2}{(s-1031)^2(s-96970)^2} - \frac{1.(-1960s^3+0.1000l^5+0.01000s^4+0.9804l^0s^2-0.1960l^0s^2)sR1}{(s-1031)^2(s-96970)^2} - \frac{10000s^2dR4}{(s-1031)^2(s-96970)^2} + \frac{0.1000l^0dG5_2}{(s-1031)^2(s-96970)^2}$$

Because the input voltage $E_i = 1.0$ V, the coefficients of the terms $dR1$, $dL2$, $dC3$, $dR4$ and $dG5_2$ from the above expression represent the sensitivities of the input admittance $Y_{1,4,1,4}$ in respect of the parameters $R1$, $L2$, $C3$, $R4$ and $G5_2$, respectively.

If the two circuits are analyzed together (Fig. 6), the independent sources from C_i become controlled sources, their controlling currents belonging to the original circuit. The solution given by SYAMNM program contains the same expressions for the variables of interest as in the case of independent solution of the two circuits.

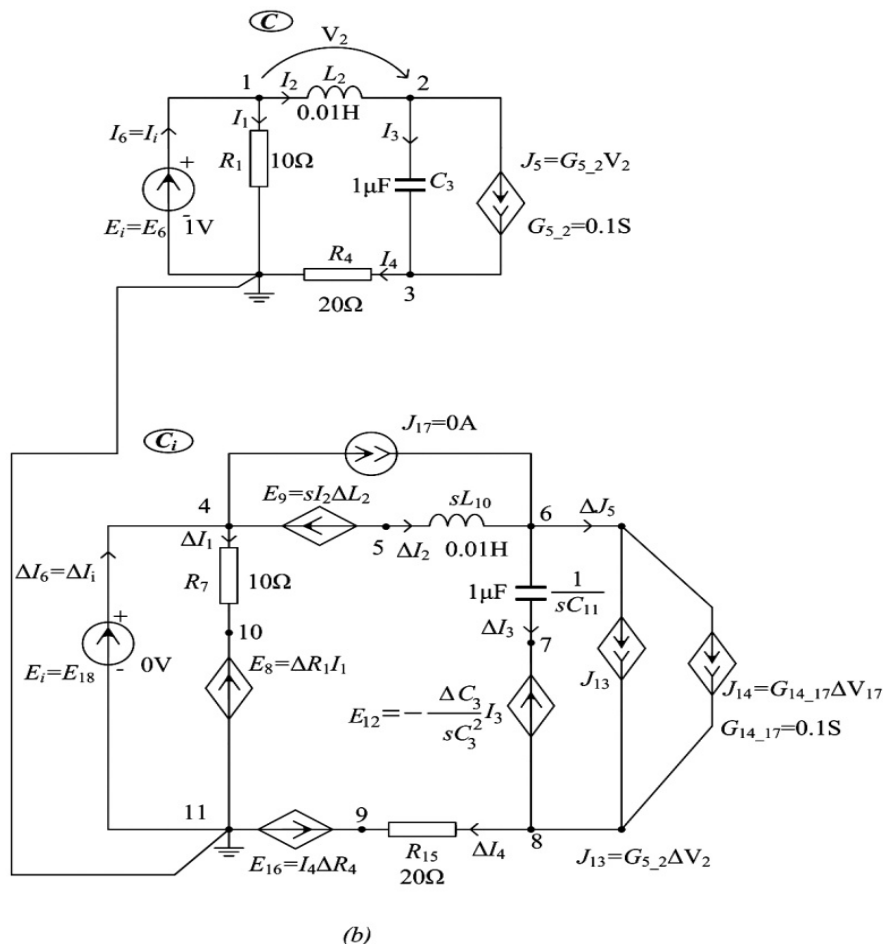


Figure 6: The two circuits connected to be solved together.

Remarks

From **Fig. 2** and from the above examples we can observe that the topology of the incremental circuit is much more complicated than of the original circuit. This increases the computational effort.

3. The Adjoint Circuit used for Sensitivity Analysis

The method for sensitivity analysis presented later is based on an auxiliary circuit with the same topology as the original one, which means they have the same incidence matrix.

It is well known that Kirchhoff's Current Law (KCL), respectively Kirchhoff's voltage law (KVL), are algebraic constraints on branch voltages, respectively branch currents, originating from the branch interconnection, while they are independent of the branch characteristics.

If we write KCL as $\mathbf{A}\mathbf{i} = \mathbf{0}$ and KVL as $\mathbf{v} = \mathbf{A}^t \mathbf{v}_n$, the expression of the total instantaneous power delivered to the circuit by the terminals of its branches is:

$$\mathbf{v}^t \mathbf{i} = (\mathbf{A}^t \mathbf{v}_n)^t \mathbf{i} = \mathbf{v}_n^t \mathbf{A} \mathbf{i} = \mathbf{0} \quad (24)$$

For any lumped circuit, (24) can be written in the form:

$$\mathbf{v}^t \mathbf{i} = \mathbf{i}^t \mathbf{v} = \mathbf{0} \quad (25)$$

called Tellegen's theorem.

The expression (25) is valid also for two circuits with the same topology or for the same circuit in two different operation behaviors, but in these cases the physical meaning of "total instantaneous power delivered to the circuit" is lost.

For two circuits, \mathcal{C} and $\hat{\mathcal{C}}$, with the same topology, (25) becomes:

$$\mathbf{v}^t \hat{\mathbf{i}} = \mathbf{i}^t \hat{\mathbf{v}} = \hat{\mathbf{v}}^t \mathbf{i} = \hat{\mathbf{i}}^t \mathbf{v} = \mathbf{0} \quad (26)$$

If we consider the perturbed circuit \mathcal{C}_p whose branch voltages and currents satisfy the equations:

$$\begin{aligned} \mathbf{v}_p &= \mathbf{v} + \Delta \mathbf{v} \\ \mathbf{i}_p &= \mathbf{i} + \Delta \mathbf{i} \end{aligned} \quad (27)$$

adapting (26) for \mathcal{C} and \mathcal{C}_p we get:

$$\begin{aligned} \hat{\mathbf{i}}^t \Delta \mathbf{v} &= \mathbf{0} \\ \hat{\mathbf{v}}^t \Delta \mathbf{i} &= \mathbf{0} \end{aligned} \quad (28)$$

equivalent to:

$$\hat{\mathbf{i}}^t \Delta \mathbf{v} - \hat{\mathbf{v}}^t \Delta \mathbf{i} = \mathbf{0} \quad (29)$$

Note. The above equations are valid not only in instantaneous values but also in Laplace transforms or phasors of currents and voltages.

To use the above theory for sensitivity analysis, the adjoint circuit of the original one has to be considered [2, 4, 6-13].

Definition

Two linear time-invariant circuits \mathcal{C} and $\hat{\mathcal{C}}$ are said to be adjoint circuits of each other if the following conditions are satisfied:

1. Both circuits have the same topology, *i.e.* $A = \hat{A}$. For the controlled sources we consider:

- the controlling voltage is that across an open-circuit branch (modeled in both circuits by an ideal independent current source having the current magnitude equal to zero);

- the controlling current is that through a short-circuit branch (modeled in both circuits by an ideal independent voltage source having the electromotive force (e.m.f.) equal to zero).

2. If the two sets of branches can be described by impedances, then the corresponding matrices satisfy the equality:

$$\mathbf{Z}_b^t = \hat{\mathbf{Z}}_b \quad (30)$$

or, if the admittances exist, then:

$$\mathbf{Y}_b^t = \hat{\mathbf{Y}}_b \quad (31)$$

Using a hybrid description of the branches [2,4,14-16] by means of hybrid matrices \mathbf{H}_b and $\hat{\mathbf{H}}_b$ respectively, we have:

$$\begin{bmatrix} \mathbf{I}_{b1} \\ \mathbf{V}_{b2} \end{bmatrix} = \begin{bmatrix} \mathbf{Y}_{b11} & \mathbf{B}_{b12} \\ \mathbf{A}_{b21} & \mathbf{Z}_{b22} \end{bmatrix} \begin{bmatrix} \mathbf{V}_{b1} \\ \mathbf{I}_{b2} \end{bmatrix} = \mathbf{H}_b \cdot \begin{bmatrix} \mathbf{V}_{b1} \\ \mathbf{I}_{b2} \end{bmatrix} \quad (32)$$

and

$$\begin{bmatrix} \hat{\mathbf{I}}_{b1} \\ \hat{\mathbf{V}}_{b2} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{Y}}_{b11} & \hat{\mathbf{B}}_{b12} \\ \hat{\mathbf{A}}_{b21} & \hat{\mathbf{Z}}_{b22} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{V}}_{b1} \\ \hat{\mathbf{I}}_{b2} \end{bmatrix} = \hat{\mathbf{H}}_b \cdot \begin{bmatrix} \hat{\mathbf{V}}_{b1} \\ \hat{\mathbf{I}}_{b2} \end{bmatrix} \quad (33)$$

where \mathbf{V}_{b1} and $\hat{\mathbf{V}}_{b1}$ (\mathbf{I}_{b2} and $\hat{\mathbf{I}}_{b2}$) are the independent voltage (current) controlled variables in the two adjoint circuits, while \mathbf{I}_{b1} and $\hat{\mathbf{I}}_{b1}$ (\mathbf{V}_{b2} and $\hat{\mathbf{V}}_{b2}$) are the complementary variables.

The condition for \mathcal{C} and $\hat{\mathcal{C}}$ to be adjoint circuits of each other is:

$$\begin{bmatrix} \hat{\mathbf{Y}}_{b11} & \hat{\mathbf{B}}_{b12} \\ \hat{\mathbf{A}}_{b21} & \hat{\mathbf{Z}}_{b22} \end{bmatrix} = \begin{bmatrix} \mathbf{Y}_{b11}^t & -\mathbf{A}_{b21}^t \\ -\mathbf{B}_{b12}^t & \mathbf{Z}_{b22}^t \end{bmatrix} \quad (34)$$

3. The independent sources in the two circuits have the same nature (voltage or current sources) but they need not have the same values.

The above definition suggests the rules to build the adjoint circuit $\hat{\mathcal{C}}$ of a given circuit \mathcal{C} as they are presented in **Fig. 7**.

In general in applications it is convenient to extract all independent sources to the ports so that a multiport remains with internal branches having $(\mathbf{v}_b, \mathbf{i}_b)$. The ports, representing the independent sources and the output variables including the voltages of the open-circuit ports and the currents of the short-circuit ports, are characterized by $(\mathbf{v}_p, \mathbf{i}_p)$.

Representation in the original circuit \mathcal{C}	Representation in the adjoint circuit $\hat{\mathcal{C}}$

Figure 7: Representation of the elements in the original and in the adjoint circuit.

Note. An open-circuit port can be modeled by an ideal independent voltage source with null e.m.f. while a short-circuit port can be modeled by an ideal independent current source with null current. In this way the connection graphs of \mathcal{C} and $\hat{\mathcal{C}}$ are identical.

With the above partition, the total number of the circuit branches is $b_t = b + p$.

Using the Laplace transforms of the voltages and currents associated to the circuits \mathcal{C} , \mathcal{C}_p and $\hat{\mathcal{C}}$, for small changes of the parameters, Eq. (29) becomes:

$$\sum_{k=1}^{b_t} [\hat{I}_k \delta V_k - \hat{V}_k \delta I_k] = 0 \quad (35)$$

The incremental changes can be made both in the internal branches and in the port variables. Identifying the contributions of the internal branches (b), independent voltage sources - including those that model the short-circuit ports (p_e), and independent current sources - including those that model the open-circuit ports (p_j), $p = p_e + p_j$, (35) becomes:

$$\sum_{k=1}^b [\hat{I}_k \delta V_k - \hat{V}_k \delta I_k] + \sum_{e=1}^{p_e} [\hat{I}_e \delta V_e - \hat{V}_e \delta I_e] + \sum_{j=1}^{p_j} [\hat{I}_j \delta V_j - \hat{V}_j \delta I_j] = 0 \quad (36)$$

In general, in order to compute the circuit function sensitivities, the variable of interest is the voltage variation at an open-circuit output port or the current variation through a short-circuit output port (see the definition of circuit functions in Chapter 4), arising from small changes in one or more parameters.

To this end an equation will be deduced from (36) having in the left side a single variable - the voltage or the current of interest, and in the right side - the terms associated with the parameters that change the values. In formulation of such an equation, the branch from \mathcal{C} , corresponding to an open-circuit port or to a short-circuit one, has to be represented in $\hat{\mathcal{C}}$ as in Fig. 8.

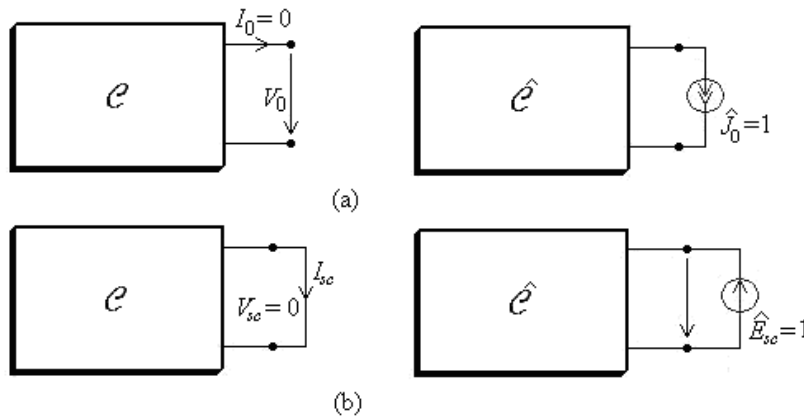


Figure 8: Correspondence between an open-circuit port (a) or a short-circuit port (b) from \mathcal{C} and the branches from $\hat{\mathcal{C}}$.

When the parameters associated to the independent sources are unchangeable, an ideal independent voltage (current) source from \mathcal{C} corresponds to a short-circuit (an open-circuit) in $\hat{\mathcal{C}}$, as in Fig. 9.

If we are interested in the voltage variation of an open-circuit port, (V_0 in Fig. 8(a)), as a result of changes in all the circuit parameters, then from (36), taking into account the conventions in Figs. 8 and 9, we get:

$$\delta V_0 = \sum_{k=1}^b [-\hat{I}_k \delta V_k] + \sum_{e=1}^{p_e} [\hat{V}_e \delta I_e - \hat{I}_e \delta V_e] + \sum_{j=1}^{p_j} [\hat{V}_j \delta I_j] \quad (37)$$

When the current variation of a short-circuit port (I_{sc} in **Fig. 8(b)**) is of interest, (36) becomes:

$$-\delta I_{sc} = \sum_{k=1}^b [-\hat{I}_k \delta V_k] + \sum_{e=1}^{p_e} [\hat{V}_e \delta I_e - \hat{I}_e \delta V_e] + \sum_{j=1}^{p_j} [\hat{V}_j \delta I_j] \quad (38)$$

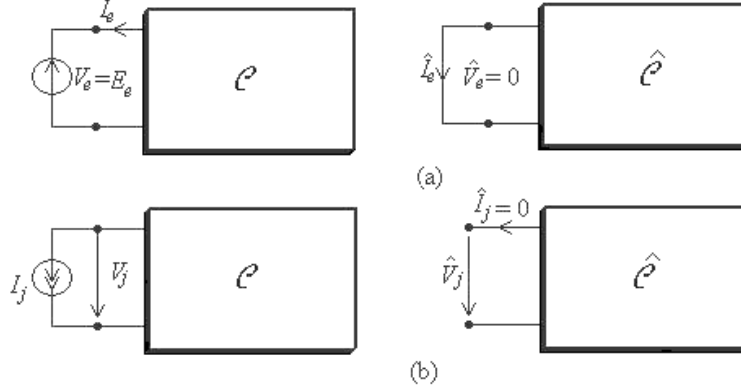


Figure 9: Correspondence between the independent voltage and current sources from \mathcal{C} and the branches from $\hat{\mathcal{C}}$.

When the independent source from the original circuit is equal to the unity, the voltage V_0 (the current I_{sc}) represents a circuit function.

For the sensitivity computation, all the terms in the right side of (37) and (38) are equal to zero, except for the terms associated to the parameters which change their values. These parameters can be the voltage V_e (the current I_j) of an independent source, R , L , C , one parameter of a controlled source, temperature *etc.*

For example in the case of V_e or I_j variation, the expressions of the voltage variation of an open-circuit port are:

$$\delta V_0 = -\hat{I}_e \delta V_e \quad \Rightarrow \quad S(V_0, V_e) = \frac{\partial V_0}{\partial V_e} = -\hat{I}_e \quad (39)$$

respectively

$$\delta V_0 = \hat{V}_j \delta I_j \quad \Rightarrow \quad S(V_0, I_j) = \frac{\partial V_0}{\partial I_j} = \hat{V}_j \quad (40)$$

Applying (35) for each circuit element, the results presented in **Table 1** are obtained.

Some situations can be discussed:

a) When branch impedance matrices of \mathcal{C} and $\hat{\mathcal{C}}$ exist, they satisfy (30) and the following relations are valid for the internal branches:

$$\begin{aligned} \mathbf{V}_b &= \mathbf{Z}_b \mathbf{I}_b \\ \hat{\mathbf{V}}_b &= \hat{\mathbf{Z}}_b \hat{\mathbf{I}}_b \end{aligned} \quad (41)$$

b) When the open-circuit impedance matrices of the two multiport exist, we can write to the ports

$$\begin{aligned} V_p &= -Z_0 I_p \\ \hat{V}_p &= -\hat{Z}_0 \hat{I}_p \end{aligned} \quad (42)$$

Writing (26) in Laplace transforms we get

$$\hat{I}^t V - \hat{V}^t I = \mathbf{0} \quad (43)$$

and taking into account the partition in internal branches and ports we rewrite as

$$\hat{I}_b^t V_b - \hat{V}_b^t I_b = -\hat{I}_p^t V_p + \hat{V}_p^t I_p \quad (44)$$

Substituting (41) and (42) in (43) we find that

$$\hat{Z}_0 = Z_0^t \quad (45)$$

c) If the short-circuit admittance matrices of the two multiport exist, in the same way we can find

$$\hat{Y}_{sc} = Y_{sc}^t \quad (46)$$

We can evaluate the effect of small changes of the Z_b matrix entries on the currents and voltages. Taking into account that

$$\delta V_b = \delta(Z_b I_b) = \delta Z_b I_b + Z_b \delta I_b \quad (47)$$

$$\delta V_p = -\delta(Z_0 I_p) = -\delta Z_0 I_p - Z_0 \delta I_p \quad (48)$$

and the relations between the matrices of the original circuit and those of the adjoint one, applying the adapted form of (44) for δV_b (δV_p) and δI_b (δI_p) we get

$$\hat{I}_p^t \delta Z_0 I_p = \hat{I}_b^t \delta Z_b I_b \quad (49)$$

Note. In a similar way we can deduce the equation

$$\hat{V}_p^t \delta Y_{sc} V_p = \hat{V}_b^t \delta Y_b V_b \quad (50)$$

Equations (49) and (50) are used for sensitivity computation.

In the particular case of the one-port, the matrix Z_0 (Y_{sc}) becomes Z_{ii} (Y_{ii}) and (49) ((50)) takes the form

$$\hat{I}_p^t \delta Z_{ii} I_p = \hat{I}_b^t \delta Z_b I_b \quad (51)$$

respectively

$$\hat{V}_p^t \delta Y_{ii} V_p = \hat{V}_b^t \delta Y_b V_b \quad (52)$$

In general Z_b and Y_b are not symmetric matrices.

Note. By a suitable choice of the excitations in \mathbf{e} and $\hat{\mathbf{e}}$, in the left side of (49) and (51) ((50) and (52)) we can have a single term δZ_{kj} or δY_{kj} , respectively.

Table 1: Contribution of the Circuit Elements to (35).

Element type	Description in \mathcal{C}	Description in $\hat{\mathcal{C}}$	$\sum(\hat{I}\delta V - \hat{V}\delta I)$
R	$V = RI$	$\hat{V} = R\hat{I}$	$\hat{I}\delta R$
G	$I = GV$	$\hat{I} = G\hat{V}$	$-\hat{V}\delta G$
Z	$V = ZI$	$\hat{V} = Z\hat{I}$	$\hat{I}\delta Z$
Y	$I = YV$	$\hat{I} = Y\hat{V}$	$-\hat{V}\delta Y$
C	$I = j\omega CV$	$\hat{I} = j\omega C\hat{V}$	$-j\omega\hat{V}\delta C$
L	$V = j\omega LI$	$\hat{V} = j\omega L\hat{I}$	$j\omega\hat{I}\delta L$
$E_c(V_c)$	$\begin{cases} V_c = A_{c_c}V_c \\ I_c = 0 \end{cases}$	$\begin{cases} \hat{I}_c = -A_{c_c}\hat{I}_c \\ \hat{V}_c = 0 \end{cases}$	$\hat{I}_c V_c \delta A_{c_c}$
$J_c(V_c)$	$\begin{cases} I_c = Y_{c_c}V_c \\ I_c = 0 \end{cases}$	$\begin{cases} \hat{I}_c = Y_{c_c}\hat{V}_c \\ \hat{I}_c = 0 \end{cases}$	$-\hat{V}_c V_c \delta Y_{c_c}$
$J_c(I_c)$	$\begin{cases} I_c = B_{c_c}I_c \\ V_c = 0 \end{cases}$	$\begin{cases} \hat{V}_c = -B_{c_c}\hat{V}_c \\ \hat{I}_c = 0 \end{cases}$	$-\hat{V}_c I_c \delta B_{c_c}$
$E_c(I_c)$	$\begin{cases} V_c = Z_{c_c}I_c \\ V_c = 0 \end{cases}$	$\begin{cases} \hat{V}_c = Z_{c_c}\hat{I}_c \\ \hat{V}_c = 0 \end{cases}$	$\hat{I}_c I_c \delta Z_{c_c}$
Ideal transformer	$\begin{cases} V_2 = nV_1 \\ I_1 = -nI_2 \end{cases}$	$\begin{cases} \hat{V}_2 = n\hat{V}_1 \\ \hat{I}_1 = -n\hat{I}_2 \end{cases}$	$(\hat{I}_2 V_1 + \hat{V}_1 I_2) \delta n$

For example, in the particular case of a circuit with four ports, if the excitations in the two adjoint circuits have the values:

$$\hat{\mathbf{I}}_p = [0, 1, 0, 0]^t \quad (53)$$

respectively

$$\mathbf{I}_p = [0, 0, 1, 0]^t \quad (54)$$

then (49) becomes

$$\delta Z_{23} = \hat{\mathbf{I}}_b^t \delta \mathbf{Z}_b \mathbf{I}_b \quad (55)$$

In order to apply (55), and calculate the partial derivatives of Z_{23} in respect of all the entries of \mathbf{Z}_b , two analyses have to be performed: the analysis of \mathcal{C} to obtain the branch currents \mathbf{I}_b , and the analysis of $\hat{\mathcal{C}}$ to obtain the vector $\hat{\mathbf{I}}_b$.

Remarks

1. If the partial derivatives $\partial \mathbf{Z}_0 / \partial x$ ($\partial Y_{sc} / \partial x$) are known, the partial derivatives of voltages (currents) $\partial V_p / \partial x$ ($\partial I_p / \partial x$) in respect with any parameter can be identified.

2. When the voltage (current) whose partial derivatives are of interest is not a port variable, a new port will be created, namely: for each voltage of interest a current port (by connection of an ideal independent current source with null current) and for each

current of interest - a voltage port (by connection of an ideal independent voltage source with null e.m.f.)

Example 3

Consider the resistive circuit represented in **Fig. 10(a)**. Let us compute by adjoint circuit method the input conductance sensitivities in respect of all circuit parameters $\partial G_{ii} / \partial G_1$, $\partial G_{ii} / \partial G_2$, $\partial G_{ii} / \partial G_3$, $\partial G_{ii} / \partial G_4$ and $\partial G_{ii} / \partial G_{5_6}$.

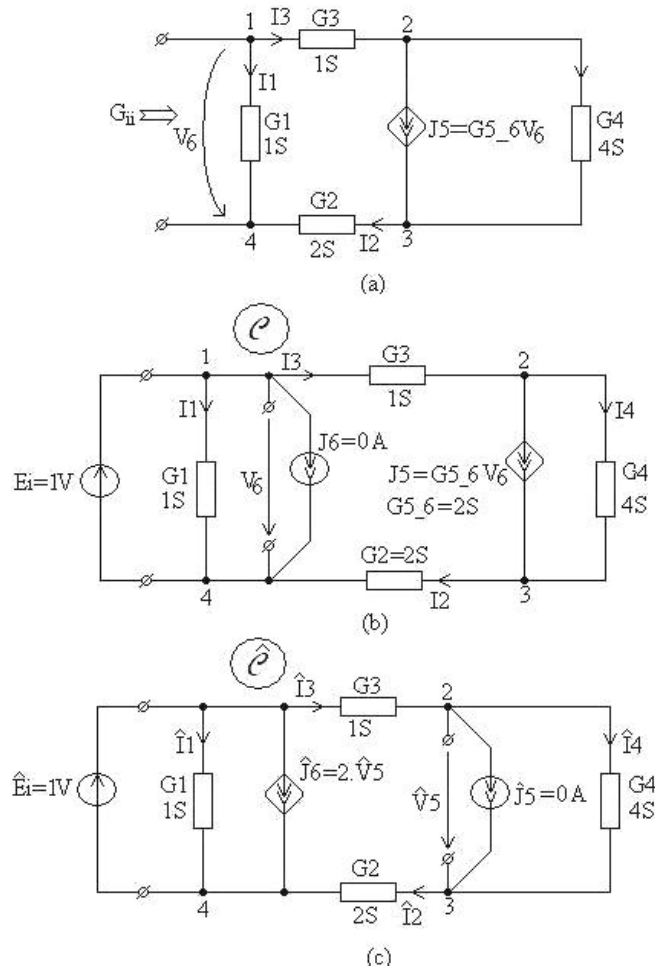


Figure 10: The original circuit (a), the multiport (b), and the adjoint circuit (c).

The original circuit is that represented in **Fig. 10(a)**. In order to identify the controlling port of the controlled source $J_5 = G_{5_6} V_6$, the original circuit is completed with a current source $J_6 = 0A$ as in **Fig. 10(b)**. The ideal independent voltage source $E_p = E_i = 1V$ is introduced in order to define the input conductance G_{ii} . The adjoint circuit $\hat{\mathcal{C}}$ is built in accordance with the **Table 1**.

Running the program, we get for the branch voltage vector in \mathcal{C} and in $\hat{\mathcal{C}}$, respectively, the following values:

$$V_b = [V_1, V_2, V_3, V_4, V_5, V_6]^t = [1, 3/7, 6/7, -2/7, -2/7, 1]^t \quad (56)$$

and

$$\hat{V}_b = [\hat{V}_1, \hat{V}_2, \hat{V}_3, \hat{V}_4, \hat{V}_5, \hat{V}_6]^t = [1, 2/7, 4/7, 1/7, 1/7, 1]^t \quad (57)$$

The branch conductance matrix \mathbf{G}_b of the circuit \mathcal{C} is:

$$\mathbf{G}_b = \begin{bmatrix} G_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & G_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & G_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & G_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & G_{5_6} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (58)$$

The small change matrix will be:

$$\delta\mathbf{G}_b = \begin{bmatrix} \delta G_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \delta G_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \delta G_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & \delta G_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \delta G_{5_6} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (59)$$

By substituting (56), (57) and (59) in (52) we get:

$$1 \cdot \delta\mathbf{G}_{ii} \cdot 1 = \begin{bmatrix} 1 & \frac{2}{7} & \frac{4}{7} & \frac{1}{7} & \frac{1}{7} & 1 \end{bmatrix} \cdot \begin{bmatrix} \delta G_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \delta G_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \delta G_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & \delta G_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \delta G_{5_6} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 3/7 \\ 6/7 \\ -2/7 \\ -2/7 \\ 1 \end{bmatrix} \quad (60)$$

Taking into account that the changes of $\delta\mathbf{G}_b$ entries are infinity small, from (60) the following sensitivities are obtained:

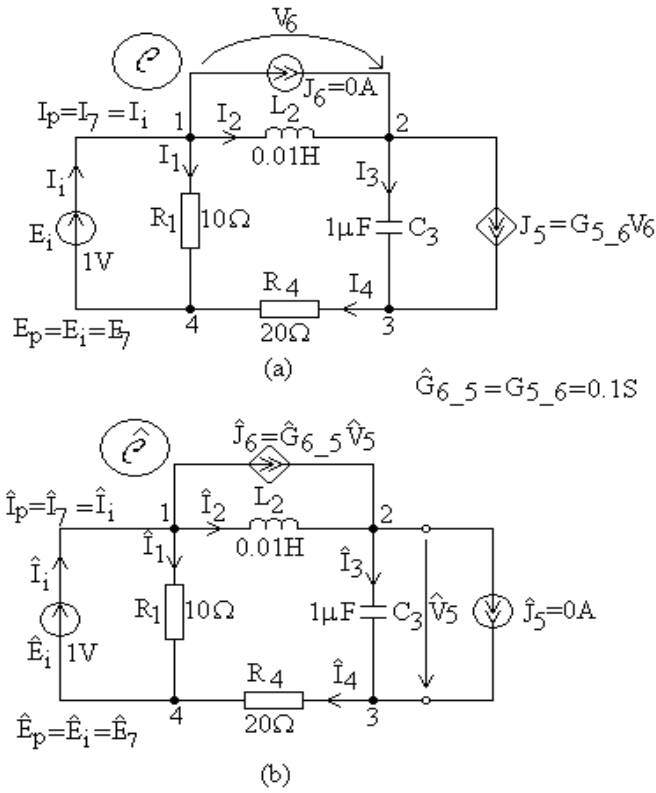
$$S_{G_{ii}} := [1.0, 0.1225, 0.49, -0.04060, 0.1429]$$

meaning that

$$\begin{aligned} \partial G_{ii} / \partial G_1 &= 1, & \partial G_{ii} / \partial G_2 &= 6/49, & \partial G_{ii} / \partial G_3 &= 24/49, \\ \partial G_{ii} / \partial G_4 &= -2/49 & \text{and} & \partial G_{ii} / \partial G_{5_6} &= 1/7 \end{aligned} \quad (61)$$

Example 4

Consider the nonreciprocal RLC circuit in **Fig. 11(a)**. The adjoint circuit is presented in **Fig. 11(b)**.



$$\hat{G}_{6_5} = G_{5_6} = 0.1S$$

Figure 11: Nonreciprocal RLC circuit.

Running the program for $E_p=1$ V, $V_b=[V_1, V_2, V_3, V_4, V_5, V_6]^t$ and $\hat{V}_b = [\hat{V}_1, \hat{V}_2, \hat{V}_3, \hat{V}_4, \hat{V}_5, \hat{V}_6]^t$ have the following expressions:

$$V_b := \begin{bmatrix} 1. \\ s^2 \\ \frac{(s - 1031.) (s - 96970.)}{-0.1000 \cdot 10^9 + 100000. s} \\ \frac{2000. s}{(s - 1031.) (s - 96970.)} \\ \frac{-0.1000 \cdot 10^9 + 100000. s}{(s - 1031.) (s - 96970.)} \\ \frac{s^2}{(s - 1031.) (s - 96970.)} \end{bmatrix} \tag{62}$$

and

$$Vb_adjt := \begin{bmatrix} 1., \frac{s(s - 100000.)}{(s - 1031.) (s - 96970.)}, \frac{0.100010^9}{(s - 1031.) (s - 96970.)}, \frac{2000. s}{(s - 1031.) (s - 96970.)}, \\ \frac{0.100010^9}{(s - 1031.) (s - 96970.)}, \frac{s(s - 100000.)}{(s - 1031.) (s - 96970.)} \end{bmatrix} \tag{63}$$

respectively.

The branch admittance matrix Y for the circuit \mathcal{C} is:

$$\mathbf{Y}_b = \begin{bmatrix} G_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{sL_2} & 0 & 0 & 0 & 0 \\ 0 & 0 & sC_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & G_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & G_{5_6} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0.1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{s \cdot 0.01} & 0 & 0 & 0 & 0 \\ 0 & 0 & s \cdot 10^{-6} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.05 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (64)$$

hence, the perturbed admittance matrix will be :

$$\begin{aligned} \delta \mathbf{Y}_b &= \begin{bmatrix} \delta G_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{sL_2^2} \delta L_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & s\delta C_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & \delta G_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \delta G_{5_6} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} = \\ &= \begin{bmatrix} \delta G_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{s \cdot 10^{-4}} \delta L_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & s\delta C_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & \delta G_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \delta G_{5_6} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{aligned} \quad (65)$$

Substituting (62)-(65) in (52) we get:

$$\begin{aligned} dY_{in} := & \left[1 \cdot dG_1 - \frac{10000 \cdot (s-100000) \cdot dL_2 \cdot s^2}{(s-1031)^2 (s-96970)^2} - \frac{0.100010^9 \cdot s \cdot dC_3 (-0.100010^9 + 100000 \cdot s)}{(s-1031)^2 (s-96970)^2} \right. \\ & \left. + \frac{0.4000 \cdot 10^7 \cdot s^2 \cdot dG_4}{(s-1031)^2 (s-96970)^2} + \frac{0.1000 \cdot 10^9 \cdot dG_{5_6} \cdot s^2}{(s-1031)^2 (s-96970)^2} \right] \end{aligned} \quad (66)$$

Taking into account that the changes of $\delta \mathbf{Y}_b$ entries are infinity small, from (66) the following sensitivities are obtained:

$$\begin{aligned} \partial Y_{ii} / \partial G_1 &= 1 \\ \partial Y_{ii} / \partial L_2 &= -\frac{10^4 s^2 (s-10^5)}{(s-1031)^2 (s-96970)^2} \\ \partial Y_{ii} / \partial C_3 &= -\frac{10^{13} s (s-10^3)}{(s-1031)^2 (s-96970)^2} \\ \partial Y_{ii} / \partial G_4 &= \frac{4 \cdot 10^6 s^2}{(s-1031)^2 (s-96970)^2} \\ \partial Y_{ii} / \partial G_{5_6} &= \frac{10^8 s^2}{(s-1031)^2 (s-96970)^2} \end{aligned} \quad (67)$$

d) The above procedure for sensitivity calculation is based on the following properties of the two multiport obtained by extracting all the independent sources to the ports, namely:

- The internal branch impedance matrix (internal branch admittance matrix) of the two multiport \mathcal{C} and $\hat{\mathcal{C}}$, \mathbf{Z}_b , respectively $\hat{\mathbf{Z}}_b$, (\mathbf{Y}_b , respectively $\hat{\mathbf{Y}}_b$), exists;
- The open-circuit impedance matrix (the short-circuit admittance matrices) of the two multiport \mathcal{C} and $\hat{\mathcal{C}}$, \mathbf{Z}_0 , respectively $\hat{\mathbf{Z}}_0$, (\mathbf{Y}_{sc} , respectively $\hat{\mathbf{Y}}_{sc}$), exists.

When the above conditions are not satisfied, the internal branches of the two multiports \mathcal{C} and $\hat{\mathcal{C}}$, can be characterized by hybrid matrices \mathbf{H}_b , respectively $\hat{\mathbf{H}}_b$ as in (32), respectively (33). For the two multiports \mathcal{C} and $\hat{\mathcal{C}}$, taking into account that the ports contain both independent voltage and current sources ($p = E + J$), the equations in hybrid matrix formulation (with \mathbf{H} respectively $\hat{\mathbf{H}}$ matrix) are:

$$\begin{bmatrix} \mathbf{I}_E \\ \mathbf{V}_J \end{bmatrix} = \begin{bmatrix} \mathbf{Y}_{EE} & \mathbf{B}_{EJ} \\ \mathbf{A}_{JE} & \mathbf{Z}_{JJ} \end{bmatrix} \begin{bmatrix} \mathbf{V}_E \\ \mathbf{I}_J \end{bmatrix} = \mathbf{H} \cdot \begin{bmatrix} \mathbf{V}_E \\ \mathbf{I}_J \end{bmatrix} \quad (68)$$

$$\begin{bmatrix} \hat{\mathbf{I}}_E \\ \hat{\mathbf{V}}_J \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{Y}}_{EE} & \hat{\mathbf{B}}_{EJ} \\ \hat{\mathbf{A}}_{JE} & \hat{\mathbf{Z}}_{JJ} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{V}}_E \\ \hat{\mathbf{I}}_J \end{bmatrix} = \hat{\mathbf{H}} \cdot \begin{bmatrix} \hat{\mathbf{V}}_E \\ \hat{\mathbf{I}}_J \end{bmatrix} \quad (69)$$

where \mathbf{V}_E and $\hat{\mathbf{V}}_E$, \mathbf{I}_J and $\hat{\mathbf{I}}_J$ are independent variables in the two adjoint circuits, while \mathbf{I}_E and $\hat{\mathbf{I}}_E$ (currents of the voltage sources), \mathbf{V}_J and $\hat{\mathbf{V}}_J$ (voltages of the current sources) are the complementary variables.

Between the hybrid matrices of the adjoint circuits the next relation is valid [3]:

$$\begin{bmatrix} \hat{\mathbf{Y}}_{EE} & \hat{\mathbf{B}}_{EJ} \\ \hat{\mathbf{A}}_{JE} & \hat{\mathbf{Z}}_{JJ} \end{bmatrix} = \begin{bmatrix} \mathbf{Y}_{EE}^t & -\mathbf{A}_{JE}^t \\ -\mathbf{B}_{EJ}^t & \mathbf{Z}_{JJ}^t \end{bmatrix} \quad (70)$$

Adapting (44) to small changes of the parameter values, we obtain:

$$\hat{\mathbf{I}}_b^t \delta \mathbf{V}_b - \hat{\mathbf{V}}_b^t \delta \mathbf{I}_b = -\hat{\mathbf{I}}_p^t \delta \mathbf{V}_p + \hat{\mathbf{V}}_p^t \delta \mathbf{I}_p \quad (71)$$

Using the port variable partition, a relation between the changes in \mathbf{H} due to the changes in \mathbf{H}_b can be derived:

$$\begin{bmatrix} \hat{\mathbf{V}}_E^t & -\hat{\mathbf{I}}_J^t \end{bmatrix} \begin{bmatrix} \delta \mathbf{Y}_{EE} & \delta \mathbf{B}_{EJ} \\ \delta \mathbf{A}_{JE} & \delta \mathbf{Z}_{JJ} \end{bmatrix} \begin{bmatrix} \mathbf{V}_E \\ \mathbf{I}_J \end{bmatrix} = \begin{bmatrix} -\hat{\mathbf{V}}_{b1}^t & \hat{\mathbf{I}}_{b2}^t \end{bmatrix} \begin{bmatrix} \delta \mathbf{Y}_{b11} & \delta \mathbf{B}_{b12} \\ \delta \mathbf{A}_{b21} & \delta \mathbf{Z}_{b22} \end{bmatrix} \begin{bmatrix} \mathbf{V}_{b1} \\ \mathbf{I}_{b2} \end{bmatrix} \quad (72)$$

Note. Developing (71) and taking into account that the independent sources are unchangeable, so that $\delta \mathbf{V}_E = 0$ and $\delta \mathbf{I}_J = 0$, we obtain (72).

Equation (72) is used for the computation of partial derivatives of interest by the following algorithm:

Step 1: Analyzing the circuit \mathcal{C} in order to obtain the vectors \mathbf{V}_{b1} and \mathbf{I}_{b2} ;

Step 2: Selecting the excitations for the adjoint circuit $\hat{\mathcal{C}}$, such that one side of (72) yields only one term that could be the derivative of an output voltage or current: δV_o , δI_o or δH_{kj} , perform the analysis of $\hat{\mathcal{C}}$ to obtain $\hat{\mathbf{V}}_{b1}^t$ ($\hat{\mathbf{V}}_{b1}^t$) and $\hat{\mathbf{I}}_{b2}^t$ ($\hat{\mathbf{I}}_{b2}^t$).

Step 3: Evaluating the right side of (72) either directly, by multiplying the matrices or simply by using the **Table 1**, the partial derivatives of interest are obtained.

Example 5

Consider the circuit in **Fig. 12**. The multiport and the adjoint circuit are presented in **Fig. 13(a, b)**. We want to compute $\delta A_{JE} = \delta A_{J_p E_p}$ in order to evaluate the voltage gain in respect of the circuit parameters.

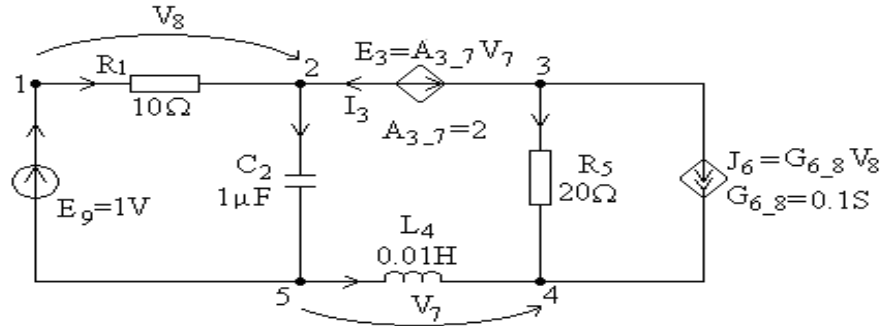


Figure 12: The nonreciprocal RLC.

We can see in (72) that to compute only δA_{JE} , the excitations have to be:

$$V_E = E_9 = 1 \text{ V}, I_J = J_{10} = 0 \text{ and } \hat{V}_E = \hat{E}_9 = 0 \text{ V}, \hat{I}_J = \hat{J}_{10} = 1 \text{ A.}$$

Step 1: Running the SYAMNM program for the circuit in **Fig. 13(a)** we obtain:

$$I_b := \begin{bmatrix} \frac{0.3333(0.3000s^2 + 200.s + 0.999910^7)}{(s + 100300)(s + 332.2)} \\ \frac{0.1000s^2}{(s + 100300)(s + 332.2)} \\ -\frac{0.3333(0.100010^8 + 200.s)}{(s + 100300)(s + 332.2)} \\ -\frac{0.3333(0.100010^8 + 200.s)}{(s + 100300)(s + 332.2)} \\ -\frac{0.1000s^2}{(s + 100300)(s + 332.2)} \\ \frac{0.3333(0.3000s^2 + 200.s + 0.999910^7)}{(s + 100300)(s + 332.2)} \\ 0. \\ 0. \end{bmatrix} \quad V_b := \begin{bmatrix} \frac{0.3333(3.s^2 + 2000.s + 0.999910^8)}{(s + 100300)(s + 332.2)} \\ \frac{100000.s}{(s + 100300)(s + 332.2)} \\ 0. \\ -\frac{0.6667s(50000.+s)}{(s + 100300)(s + 332.2)} \\ -\frac{2.000s^2}{(s + 100300)(s + 332.2)} \\ -\frac{2.000s^2}{(s + 100300)(s + 332.2)} \\ -\frac{0.6667s(50000.+s)}{(s + 100300)(s + 332.2)} \\ \frac{0.3333(3.s^2 + 2000.s + 0.999910^8)}{(s + 100300)(s + 332.2)} \end{bmatrix}$$

where

$$I_b = [I_1, I_2, I_3, I_4, I_5, J_6, J_7, J_8]^t \text{ and } V_b = [V_1, V_2, V_3, V_4, V_5, V_6, V_7, V_8]^t.$$

Step 2: If the circuit in **Fig. 13(b)** is analysed, the results are:

Step 3: Evaluating the right side of (72) by using the relations from **Table 1**, the following expression is obtained:

$$\delta A_{JE} = I_1 \hat{I}_1 \delta R_1 - s V_2 \hat{V}_2 \delta C_2 + V_7 \hat{I}_3 \delta A_{3_7} + s I_4 \hat{I}_4 \delta L_4 + I_5 \hat{I}_5 \delta R_5 - V_8 \hat{V}_6 \delta G_{6_8}.$$

Using this expression and the values computed in steps 1 and 2, we get:

$$dAEJ := - \frac{0.1111(0.3000s^2 + 200.s + 0.999910^7) (0.200010^9 + 600000s) dR1}{(s + 100300)^2 (s + 332.2)^2} \\ - \frac{33330.s^2 (0.200010^{10} + 0.600010^7 s) dC2}{(s + 100300)^2 (s + 332.2)^2} - \frac{444.5s^2 (50000.+s) dA3_7}{(s + 100300)^2 (s + 332.2)^2} \\ - \frac{666.6s^2 (0.100010^8 + 200.s) dL4}{(s + 100300)^2 (s + 332.2)^2} - \frac{0.1000(s + 99670.) (s + 334.5) s^2 dR5}{(s + 100300)^2 (s + 332.2)^2} \\ - \frac{6.666(s + 99670.) (s + 334.5) (3.s^2 + 2000.s + 0.999910^8) dG6_8}{(s + 100300)^2 (s + 332.2)^2}$$

The partial derivatives of the voltage gain $A_{3,4_1,5}$ in respect of the circuit parameters are the coefficients of the derivatives of these parameters in the above expression.

To verify the accuracy of the results obtained by the adjoint circuit method, we generate the voltage gain in symbolic form [17,18].

$$A_{3_4_1_5} := - \frac{((L4 G6_8 C2 R1 A3_7 + L4 G6_8 C2 R1) s^2 + G6_8 R1 - 1.) R5}{(L4 C2 R1 A3_7 + L4 C2 R1) s^2 + (C2 R5 R1 + L4 A3_7 + L4) s + R5 - 1. G6_8 R5 R1 + R1}$$

and derivate this expression in respect of the circuit parameters: $R1$, $C2$, $A3_7$, $L4$, $R5$, $G6_8$. After numeric substitution, we obtain the results presented in **Table 2**.

Table 2: Sensitivity of the Voltage Gain.

Results Obtained by the Adjoint Circuit Method	Results Obtained by Symbolic Analysis
$- \frac{20000.(s + 333.4) (s^2 + 666.6s + 0.333210^8)}{(s + 100300)^2 (s + 332.2)^2} \\ - \frac{0.200010^{12} (s + 333.3) s^2}{(s + 100300)^2 (s + 332.2)^2} \\ - \frac{444.5s^2 (50000.+s)}{(s + 100300)^2 (s + 332.2)^2} \\ - \frac{133300.s^2 (50010.+s)}{(s + 100300)^2 (s + 332.2)^2} \\ - \frac{0.1000(s + 99670.) (s + 334.5) s^2}{(s + 100300)^2 (s + 332.2)^2} \\ - \frac{20.00(s + 334.3) (s + 99650.) (s^2 + 666.6s + 0.333510^8)}{(s + 100300)^2 (s + 332.2)^2}$	$- \frac{20000.(s + 333.3) (s^2 + 666.6s + 0.333310^8)}{(s + 100300)^2 (s + 332.2)^2} \\ - \frac{0.200010^{12} (s + 333.3) s^2}{(s + 100300)^2 (s + 332.2)^2} \\ - \frac{444.4s^2 (50000.+s)}{(s + 100300)^2 (s + 332.2)^2} \\ - \frac{133300.s^2 (50000.+s)}{(s + 100300)^2 (s + 332.2)^2} \\ - \frac{0.1000(s + 99670.) (s + 334.5) s^2}{(s + 100300)^2 (s + 332.2)^2} \\ - \frac{20.00(s + 99670.) (s + 334.5) (s^2 + 666.6s + 0.333310^8)}{(s + 100300)^2 (s + 332.2)^2}$

Example 6

To illustrate the generality of the adjoint circuit method we consider the linear circuit represented in **Fig. 14**, containing all the four type of controlled sources. The multiport and the adjoint circuit built in accordance with **Table 1** are done in **Fig. 15(a, b)**. We want to compute the partial derivatives of δA_{JE} in respect of the circuit parameters.

By examining (72) we see that in order to compute only δA_{JE} we have to choose the following combinations: $V_E = E_{15} = 1$ V, $I_J = J_{13} = 0$ A and $V_E = E_{15} = 0$ V, $I_J = J_{13} = -1$ A.

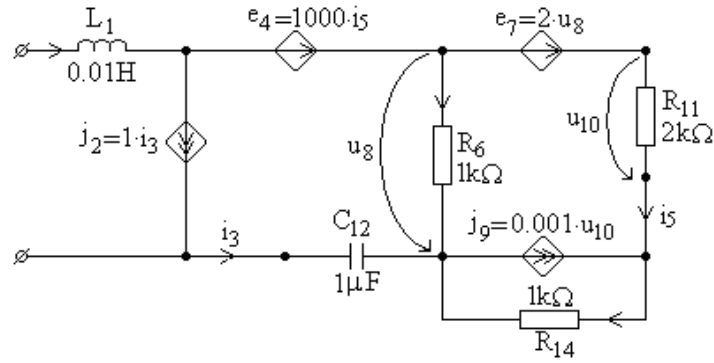


Figure 14: The original circuit.

Step 1: Running the SYAMNM program for the circuit in Fig. 15(a) we get:

$$I_b^t = [I_1 \quad J_2 \quad I_3 \quad I_4 \quad I_5 \quad I_6 \quad I_7 \quad J_8 \quad J_9 \quad J_{10} \quad I_{11} \quad I_{12} \quad I_{14}] =$$

$$\left[0, -\frac{0.004000s}{4000.+s}, -\frac{0.004000s}{4000.+s}, -\frac{0.004000s}{4000.+s}, \frac{0.001500s}{4000.+s}, \frac{0.002500s}{4000.+s}, -\frac{0.001500s}{4000.+s}, 0, \frac{0.003000s}{4000.+s}, 0, \frac{0.001500s}{4000.+s}, \frac{0.004000s}{4000.+s}, \frac{0.004500s}{4000.+s} \right]$$

$$V_b^t = [V_1 \quad V_2 \quad E_3 \quad E_4 \quad E_5 \quad V_6 \quad E_7 \quad V_8 \quad V_9 \quad V_{10} \quad V_{11} \quad V_{12} \quad V_{14}] =$$

$$\left[0, 1, 0, \frac{1.500s}{4000.+s}, 0, \frac{2.500s}{4000.+s}, \frac{5.s}{4000.+s}, \frac{2.500s}{4000.+s}, -\frac{4.500s}{4000.+s}, \frac{3.s}{4000.+s}, \frac{3.s}{4000.+s}, \frac{4000.}{4000.+s}, \frac{4.500s}{4000.+s} \right]$$

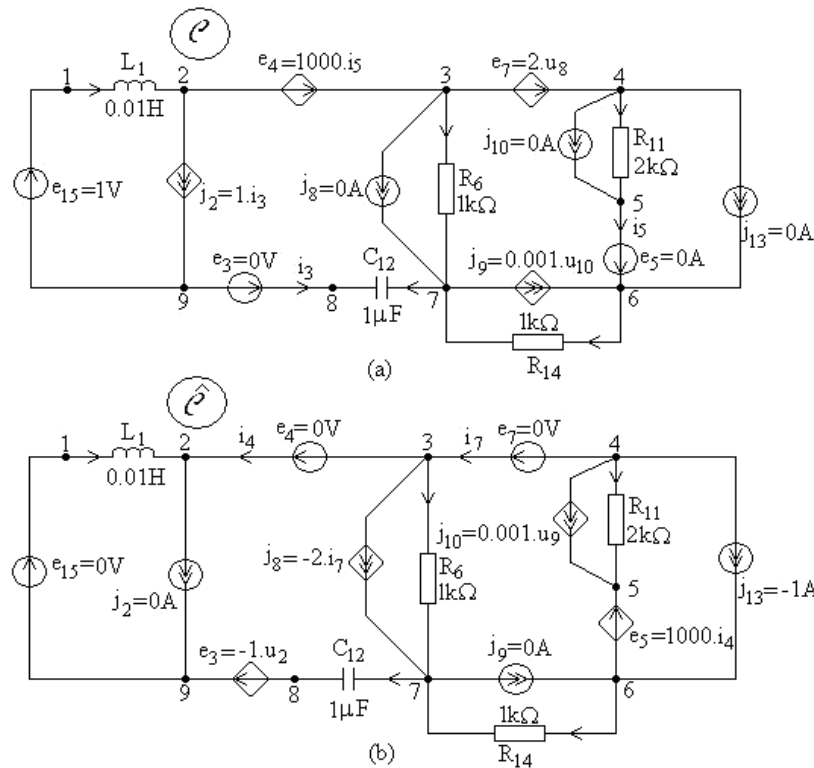


Figure 15: The multiport (a) and the adjoint circuit (b).

Step 2: Running the SYAMNM program for the circuit in Fig. 15(b) we get:

$$\hat{I}_b^t = [\hat{I}_1 \quad \hat{J}_2 \quad \hat{I}_3 \quad \hat{I}_4 \quad \hat{I}_5 \quad \hat{I}_6 \quad \hat{I}_7 \quad \hat{J}_8 \quad \hat{J}_9 \quad \hat{J}_{10} \quad \hat{I}_{11} \quad \hat{I}_{12} \quad \hat{I}_{14}] =$$

$$\begin{aligned} & \left[-\frac{3.s}{4000.+s}, 0, \frac{3.s}{4000.+s}, \frac{3.s}{4000.+s}, \frac{3000.}{4000.+s}, \frac{3000.}{4000.+s}, \frac{1000.+s}{4000.+s}, -\frac{2.(1000.+s)}{4000.+s}, 0, \frac{1000.+s}{4000.+s}, -\frac{1.(s-2000)}{4000.+s}, -\frac{3.s}{4000.+s}, \right. \\ & \left. -\frac{1.(1000.+s)}{4000.+s} \right] \\ \hat{V}_b^t &= \left[\hat{V}_1 \quad \hat{V}_2 \quad \hat{E}_3 \quad \hat{E}_4 \quad \hat{E}_5 \quad \hat{V}_6 \quad \hat{E}_7 \quad \hat{V}_8 \quad \hat{V}_9 \quad \hat{V}_{10} \quad \hat{V}_{11} \quad \hat{V}_{12} \quad \hat{V}_{14} \right] = \\ & \left[-\frac{0.03000s^2}{4000.+s}, \frac{0.03000s^2}{4000.+s}, -\frac{0.03000s^2}{4000.+s}, 0, \frac{3000.s}{4000.+s}, \frac{0.300010^7}{4000.+s}, 0, \frac{0.300010^7}{4000.+s}, \frac{1000.(1000.+s)}{4000.+s}, \frac{2000.(s-2000.)}{4000.+s}, \right. \\ & \left. -\frac{2000.(s-2000.)}{4000.+s}, -\frac{0.300010^7}{4000.+s}, -\frac{1000.(1000.+s)}{4000.+s} \right] \end{aligned}$$

Step 3. Evaluating the right side of (72) by using the relations in **Table 1**, we get:

$$\begin{aligned} \delta A_{JE} &= sI_1 \hat{I}_1 \delta L_1 - \hat{V}_2 I_3 \delta B_{2_3} + I_5 \hat{I}_4 \delta R_{4_5} + I_6 \hat{I}_6 \delta R_6 + V_8 \hat{I}_7 \delta A_{7_8} - \\ &- V_{10} \hat{V}_9 \delta G_{9_10} + I_{11} \hat{I}_{11} \delta R_{11} - sV_{12} \hat{V}_{12} \delta C_{12} + I_{14} \hat{I}_{14} \delta R_{14}. \end{aligned}$$

Using the values computed in the steps 1 and 2, it results:

$$\begin{aligned} dA_{JE} &= \frac{0.0001200s^3 dB_{2_3}}{(4000.+s)^2} + \frac{0.004500s^2 dR_{4_5}}{(4000.+s)^2} + \frac{7.500s dR_6}{(4000.+s)^2} + \frac{2.500(1000.+s) s dA_{7_8}}{(4000.+s)^2} - \frac{0.001500(s-2000.) s dR_{11}}{(4000.+s)^2} + \frac{0.120010^{11} s dC_{12}}{(4000.+s)^2} \\ &- \frac{0.004500(1000.+s) s dR_{14}}{(4000.+s)^2} \end{aligned}$$

The partial derivatives of the voltage gain $A_{4,6,1,9}$ in respect of the circuit parameters are the coefficients of the derivatives of these parameters in the above expression.

In order to verify the results obtained by the adjoint circuit method, we first generate in symbolic form the voltage gain $A_{4,6,1,9} = A_{JE}$

$$\begin{aligned} A_{JE} &= -s C12(A7_8+1.)R6R11((-1.L1C12R14-1.L1C12A7_8R6+L1B2_3C12R11+L1B2_3C12R14-1.L1C12R6 \\ &-1.L1C12G9_10R11R14-1.L1C12R11+L1G9_10B2_3C12R11R14+L1A7_8B2_3C12R6+L1B2_3C12R6)s^2 \\ &+(C12R4_5R6+C12R4_5A7_8R6-1.C12R6R14-1.G9_10C12R6R11R14-1.C12R6R11)s-1.A7_8R6-1.G9_10R11R14-1.R6 \\ &-1.R14-1.R11) \end{aligned}$$

Deriving that expression in respect of the circuit parameters in **Fig. 14**: L1, B2_3, R4_5, R6, A7_8, G9_10, R11, C12, R14, and substituting the numeric nominal values, we obtain the same results as we present in **Table 3**.

Remarks.

From the point of view of the computation effort, the incremental circuit method for sensitivity computation is less efficient than the adjoint circuit method, because of the complexity of the incremental circuit as we can see comparing the **Fig. 2** and the **Fig. 7**. Anyway, the incremental circuit method has the following advantages:

- 1) Using the ideal independent voltage and current sources $I_b \delta R_b$, respectively $V_b \delta G_b$ we can better understand the effects of changes in branch resistances and conductances;
- 2) The values of branch voltages and currents of the incremental circuit δI_b and δV_b , obtained by the incremental circuit analysis are, often used in applications.

4. Sensitivity Analysis by Bykhovski Method

Developed by Bykhovski [2] and described in detail by Perkins and Cruz [8, 9], this is also a method for sensitivity computation by auxiliary circuits. It will be presented later.

Table 3: Sensitivity by the Two Methods.

Results Obtained by the Adjoint Circuit Method.	Results Obtained by Symbolic Analysis.
$\begin{aligned} & 0. \\ & \frac{0.0001200 s^3}{(4000. + s)^2} \\ & \frac{0.004500 s^2}{(4000. + s)^2} \\ & \frac{7.500 s}{(4000. + s)^2} \\ & \frac{2.500 (1000. + s) s}{(4000. + s)^2} \\ & - \frac{3000. (1000. + s) s}{(4000. + s)^2} \\ & - \frac{0.001500 s (s - 2000.)}{(4000. + s)^2} \\ & \frac{0.1200 10^{11} s}{(4000. + s)^2} \\ & - \frac{0.004500 (1000. + s) s}{(4000. + s)^2} \end{aligned}$	$\begin{aligned} & 0 \\ & \frac{0.0001200 s^3}{(4000. + s)^2} \\ & \frac{0.004500 s^2}{(4000. + s)^2} \\ & \frac{7.500 s}{(4000. + s)^2} \\ & \frac{2.500 (1000. + s) s}{(4000. + s)^2} \\ & - \frac{3000. (1000. + s) s}{(4000. + s)^2} \\ & - \frac{0.001500 s (s - 2000.)}{(4000. + s)^2} \\ & \frac{0.1200 10^{11} s}{(4000. + s)^2} \\ & - \frac{0.004500 (1000. + s) s}{(4000. + s)^2} \end{aligned}$

Kirchhoff's laws in complex variables or in complex frequency domain have the form,

$$\text{KCL: } \underline{AI} = \mathbf{0}; \quad \text{KVL: } \underline{BV} = \mathbf{0} \quad (73)$$

respectively

$$\text{KCL: } \underline{AI}(s) = \mathbf{0}; \quad \text{KVL: } \underline{BV}(s) = \mathbf{0} \quad (74)$$

The partial derivatives in respect of the parameter x , of (73) and (74) are:

$$\text{KCL: } \underline{A} \frac{\partial \underline{I}}{\partial x} = \mathbf{0}; \quad \text{KVL: } \underline{B} \frac{\partial \underline{V}}{\partial x} = \mathbf{0} \quad (75)$$

Respectively

$$\text{KCL: } \underline{A} \frac{\partial \underline{I}(s)}{\partial x} = \mathbf{0}; \quad \text{KVL: } \underline{B} \frac{\partial \underline{V}(s)}{\partial x} = \mathbf{0} \quad (76)$$

The only entries that are zero in the above expressions are those corresponding to the ideal independent voltage, respectively current sources, which are unchangeable. Equations (76) can be associated to an auxiliary circuit \mathcal{C}_a which has the same connection graph as the original circuit \mathcal{C} , but in which the independent sources are made passive (the voltage sources by short-circuits and the current sources by open-circuits) and the branch voltages and currents are the derivatives $\partial v_k / \partial x$, respectively $\partial i_k / \partial x$, $k = \overline{1, b}$. The auxiliary circuit built in this way is called the *sensitivity circuit*.

Consider an *RLC* linear circuit containing n_R resistors, n_C capacitors, n_L inductors, all the four type of controlled sources, and independent voltage and current sources.

The constitutive equations in dynamic behavior for the passive circuit elements take the form:

$$\begin{aligned} u_{R_k} &= R_k i_{R_k}, \quad k = \overline{1, n_R} \\ i_{C_j} &= C_j \frac{du_{C_j}}{dt}, \quad j = \overline{1, n_C} \\ u_{L_p} &= L_p \frac{di_{L_p}}{dt}, \quad p = \overline{1, n_L} \end{aligned} \quad (77)$$

If we want to compute the sensitivities in respect of the resistance R_k , then the passive element equations, have in the auxiliary circuit, the form:

$$\frac{\partial u_{R_k}}{\partial R_k} = i_{R_k} + R_k \frac{\partial i_{R_k}}{\partial R_k}, \quad \frac{\partial i_{C_j}}{\partial R_k} = C_j \frac{d}{dt} \left(\frac{\partial u_{C_j}}{\partial R_k} \right), \quad \frac{\partial u_{L_p}}{\partial R_k} = L_p \frac{d}{dt} \left(\frac{\partial i_{L_p}}{\partial R_k} \right) \quad (78)$$

The equations corresponding to small changes in circuit parameters and the branch representation in the auxiliary circuit are done in **Fig. 16**.

Example 7

Consider the nonreciprocal linear circuit in **Fig. 17(a)**, and the auxiliary (sensitivity) circuit in **Fig. 17(b)**, built according to the **Fig.16** for the computation of the sensitivities in respect to C_2 . Because the auxiliary circuit contains controlled sources whose controlling variables are located in the original circuit, the two circuits are analyzed together (having the reference node in common).

The independent voltage sources $e_7 = e_{15} = 1$ V and the independent current sources $j_6 = j_{14} = 0$ A are introduced in order to compute directly the voltage gain $A_{3,7,1,7} = V_6(s)/1.0 = V_6(s)$.

The branch current vector and the branch voltage vector of the whole circuit have the following structures:

$$\begin{aligned} \mathbf{I}_b &= [I_1, I_2, I_3, J_4, I_5, J_6, I_7, I_8, I_9, J_{10}, I_{11}, J_{12}, I_{13}, J_{14}, I_{15}]^t, \\ \mathbf{V}_b &= [V_1, V_2, V_3, V_4, V_5, V_6, V_7, V_8, V_9, V_{10}, V_{11}, V_{12}, V_{13}, V_{14}, V_{15}]^t. \end{aligned}$$

Voltage V_{14} represents the sensitivity of V_6 (and of the voltage gain $A_{3,7,1,7}$) in respect of the parameter C_2 . Performing the analysis of the whole circuit connected by the node 7, we get:

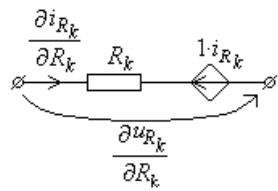
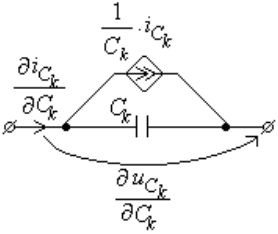
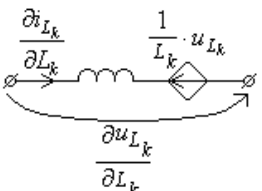
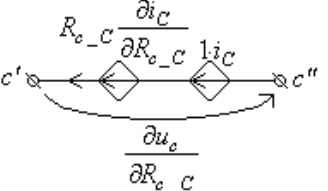
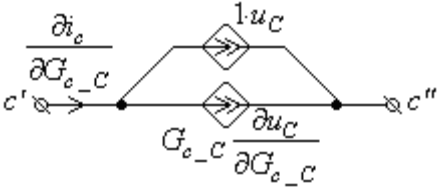
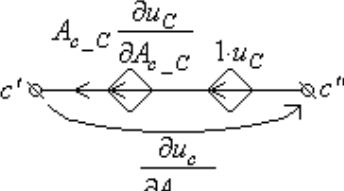
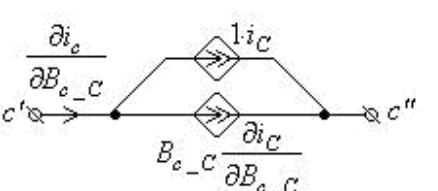
Characteristic equation	Characteristic equation in the sensitivity circuit	Branch structure in the sensitivity circuit
Resistor R_k : $u_{R_k} = R_k i_{R_k}$	$\frac{\partial u_{R_k}}{\partial R_k} = 1 \cdot i_{R_k} + R_k \frac{\partial i_{R_k}}{\partial R_k}$	
Capacitor C_k : $i_{C_k} = C_k \frac{du_{C_k}}{dt}$	$\frac{\partial i_{C_k}}{\partial C_k} = \frac{1}{C_k} \cdot i_{C_k} + C_k \frac{d}{dt} \left(\frac{\partial u_{C_k}}{\partial C_k} \right)$	
Inductor L_k : $u_{L_k} = L_k \frac{di_{L_k}}{dt}$	$\frac{\partial u_{L_k}}{\partial L_k} = \frac{1}{L_k} \cdot u_{L_k} + L_k \frac{d}{dt} \left(\frac{\partial i_{L_k}}{\partial L_k} \right)$	
$e_c(i_C)$: $u_c = e_c = R_{c_c} i_C$	$\frac{\partial u_c}{\partial R_{c_c}} = \frac{\partial e_c}{\partial R_{c_c}} = 1 \cdot i_C + R_{c_c} \frac{\partial i_C}{\partial R_{c_c}}$	
$j_c(u_C)$: $i_c = j_c = G_{c_c} u_C$	$\frac{\partial i_c}{\partial G_{c_c}} = \frac{\partial j_c}{\partial G_{c_c}} = 1 \cdot u_C + G_{c_c} \frac{\partial u_C}{\partial G_{c_c}}$	
$e_c(u_C)$: $u_c = e_c = A_{c_c} u_C$	$\frac{\partial u_c}{\partial A_{c_c}} = \frac{\partial e_c}{\partial A_{c_c}} = 1 \cdot u_C + A_{c_c} \frac{\partial u_C}{\partial A_{c_c}}$	
$j_c(i_C)$: $i_c = j_c = B_{c_c} i_C$	$\frac{\partial i_c}{\partial B_{c_c}} = \frac{\partial j_c}{\partial B_{c_c}} = 1 \cdot i_C + B_{c_c} \frac{\partial i_C}{\partial B_{c_c}}$	

Figure 16: Element representation in the auxiliary circuit.

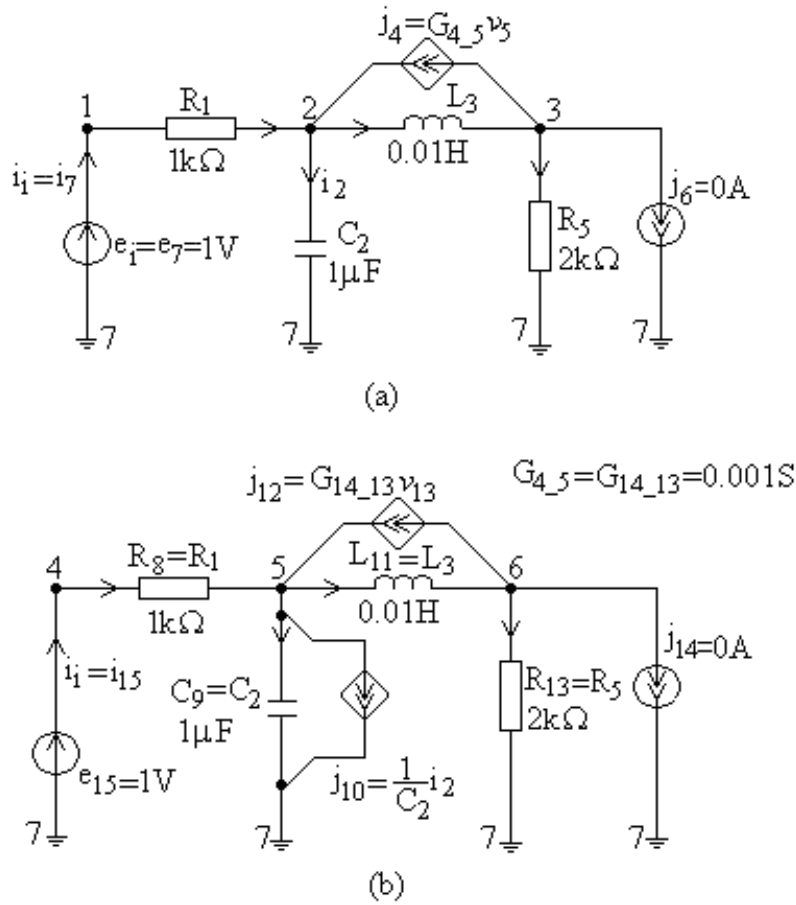


Figure 17: The multiport (a) and the sensitivity circuit (b) connected to a common node.

$$I_b := \begin{bmatrix} \frac{0.001 (s + 66162.859) (s + 503.80733)}{(s + 66155.067) (s + 1511.6000)} \\ \frac{0.001 (s + 66666.667) s}{(s + 66155.067) (s + 1511.6000)} \\ \frac{100000.00}{(s + 66155.067) (s + 1511.6000)} \\ \frac{66666.667}{(s + 66155.067) (s + 1511.6000)} \\ \frac{33333.333}{(s + 66155.067) (s + 1511.6000)} \\ 0. \\ \frac{0.001 (s + 66162.859) (s + 503.80733)}{(s + 66155.067) (s + 1511.6000)} \\ \frac{0.10000000 10^7 (s + 66666.667)^2 s}{(s + 66155.067)^2 (s + 1511.6000)^2} \\ \frac{1000.0000 (s + 66666.667)^2 s^2}{(s + 66155.067)^2 (s + 1511.6000)^2} \\ \frac{1000.0000 (s + 66666.667) s}{(s + 66155.067) (s + 1511.6000)} \\ \frac{0.1 10^{12} (s + 66666.667) s}{(s + 66155.067)^2 (s + 1511.6000)^2} \\ \frac{0.66666667 10^{11} (s + 66666.667) s}{(s + 66155.067)^2 (s + 1511.6000)^2} \\ \frac{0.33333333 10^{11} (s + 66666.667) s}{(s + 66155.067)^2 (s + 1511.6000)^2} \\ 0. \\ \frac{0.10000000 10^7 (s + 66666.667)^2 s}{(s + 66155.067)^2 (s + 1511.6000)^2} \end{bmatrix}$$

$$V_b := \begin{bmatrix} \frac{1.0 (s + 66162.859) (s + 503.80733)}{(s + 66155.067) (s + 1511.6000)} \\ \frac{0.33333333 (0.20000000 10^9 + 3000. s)}{(s + 66155.067) (s + 1511.6000)} \\ \frac{1000.0000 s}{(s + 66155.067) (s + 1511.6000)} \\ \frac{1000.0000 s}{(s + 66155.067) (s + 1511.6000)} \\ \frac{0.66666667 10^8}{(s + 66155.067) (s + 1511.6000)} \\ \frac{0.66666667 10^8}{(s + 66155.067) (s + 1511.6000)} \\ \frac{-1.}{(s + 66155.067) (s + 1511.6000)} \\ \frac{0.1 10^{10} (s + 66666.667)^2 s}{(s + 66155.067)^2 (s + 1511.6000)^2} \\ \frac{0.1 10^{10} (s + 66666.667)^2 s}{(s + 66155.067)^2 (s + 1511.6000)^2} \\ \frac{0.1 10^{10} (s + 66666.667)^2 s}{(s + 66155.067)^2 (s + 1511.6000)^2} \\ \frac{0.1 10^{10} (s + 66666.667) s^2}{(s + 66155.067)^2 (s + 1511.6000)^2} \\ \frac{0.1 10^{10} (s + 66666.667) s^2}{(s + 66155.067)^2 (s + 1511.6000)^2} \\ \frac{0.66666667 10^{14} (s + 66666.667) s}{(s + 66155.067)^2 (s + 1511.6000)^2} \\ \frac{0.66666667 10^{14} (s + 66666.667) s}{(s + 66155.067)^2 (s + 1511.6000)^2} \\ 0. \end{bmatrix}$$

Identifying the voltage V_{14} in the above V_b vector we take out

$$V_{14} = \frac{\partial V_6}{\partial C_2} = \frac{\partial A_{3,7-1,7}}{\partial C_2} = - \frac{0.66666667 \cdot 10^{14} (s + 66666.667) s}{(s + 66155.067)^2 (s + 1511.6000)^2} \quad (79)$$

In order to verify this result, the voltage gain in symbolic form was obtained by SYAMNM program and the sensitivity $S_{C_2}^{A_{3,7-1,7}}$ was obtained as:

$$SV6_C2 := - \frac{R5((L3G4_5R1R5+L3R1)s^2+sR1R5)}{((L3C2G4_5R1R5+C2L3R1)s^2+(C2R1R5+L3+G4_5L3R5)s+R5+R1)^2} \quad (80)$$

Substituting the parameter values in (80) the same value as in (79) is obtained:

$$SA_{3_7_1_7_C2} := - \frac{0.66666667 \cdot 10^{14} (s + 66666.667) s}{(s + 66155.067)^2 (s + 1511.6000)^2} \quad (81)$$

Example 8

Consider the universal filter in Chapter 4 shown in **Fig. 10**. The four opamps are considered linear, with the input resistance $R_i = 2 \cdot 10^6 \Omega$ and the voltage gain $A = 2 \cdot 10^5$. In accordance with **Fig. 16**, the auxiliary circuit corresponding to the sensitivity computation in respect of the capacitor C_5 , is built (see **Fig. 18**).

Because the controlling variables of the controlled sources in the auxiliary circuit belong to the original one, the two circuits are analysed together, being connected by the datum node.

The independent voltage sources $e_{21} = 1 \text{ V}$ and $e_{47} = 0 \text{ V}$, and the independent current sources $j_{22} = j_{23} = j_{24} = j_{25} = j_{48} = j_{49} = j_{50} = j_{51} = 0 \text{ A}$ are introduced in order to directly compute the voltage gains of the four type of filter: $A_{4,11-1,21} = V_{22}(s)/1.0 = V_{22}(s)$, $A_{6,11-1,21} = V_{23}(s)/1.0 = V_{23}(s)$, $A_{8,21-1,21} = V_{24}(s)/1.0 = V_{24}(s)$, $A_{10,21-1,21} = V_{25}(s)/1.0 = V_{25}(s)$, and their sensitivities in respect of the capacitor C_5 : $SA_{4,21-1,21-C_5} = U_{48}(s)$; $SA_{6,21-1,21-C_5} = U_{49}(s)$; $SA_{8,21-1,21-C_5} = U_{50}(s)$ and $SA_{10,17-1,21-C_5} = U_{51}(s)$.

Running SYAMNM program we get:

$$\begin{aligned} SA_HPF_C5 &:= \frac{0.6722 \cdot 10^{13} (s + 13640.) (s + 0.2525) s}{(s^2 + 18330. s + 0.2499 \cdot 10^9)^2} & SA_LPF_C5 &:= - \frac{0.9164 \cdot 10^{18} (s + 0.2525) s}{(s^2 + 18330. s + 0.2499 \cdot 10^9)^2} \\ SA_PBF_C5 &:= \frac{0.1833 \cdot 10^{14} s (s^2 + 0.5048 s + 0.06375)}{(s^2 + 18330. s + 0.2499 \cdot 10^9)^2} & SA_RBF_C5 &:= \frac{0.9164 \cdot 10^{19} (s + 0.2525) s}{(s^2 + 18330. s + 0.2499 \cdot 10^9)^2} \end{aligned}$$

Remarks

The high order derivatives can also be computed by Bykhovski – Perkins – Cruz method. To this end high order sensitivity circuits will be attached to those of low order. The method can be extended to sensitivity computation of the nonlinear analog circuits. The major drawback of the method consists in limitation of sensitivity computation in respect to only one parameter.

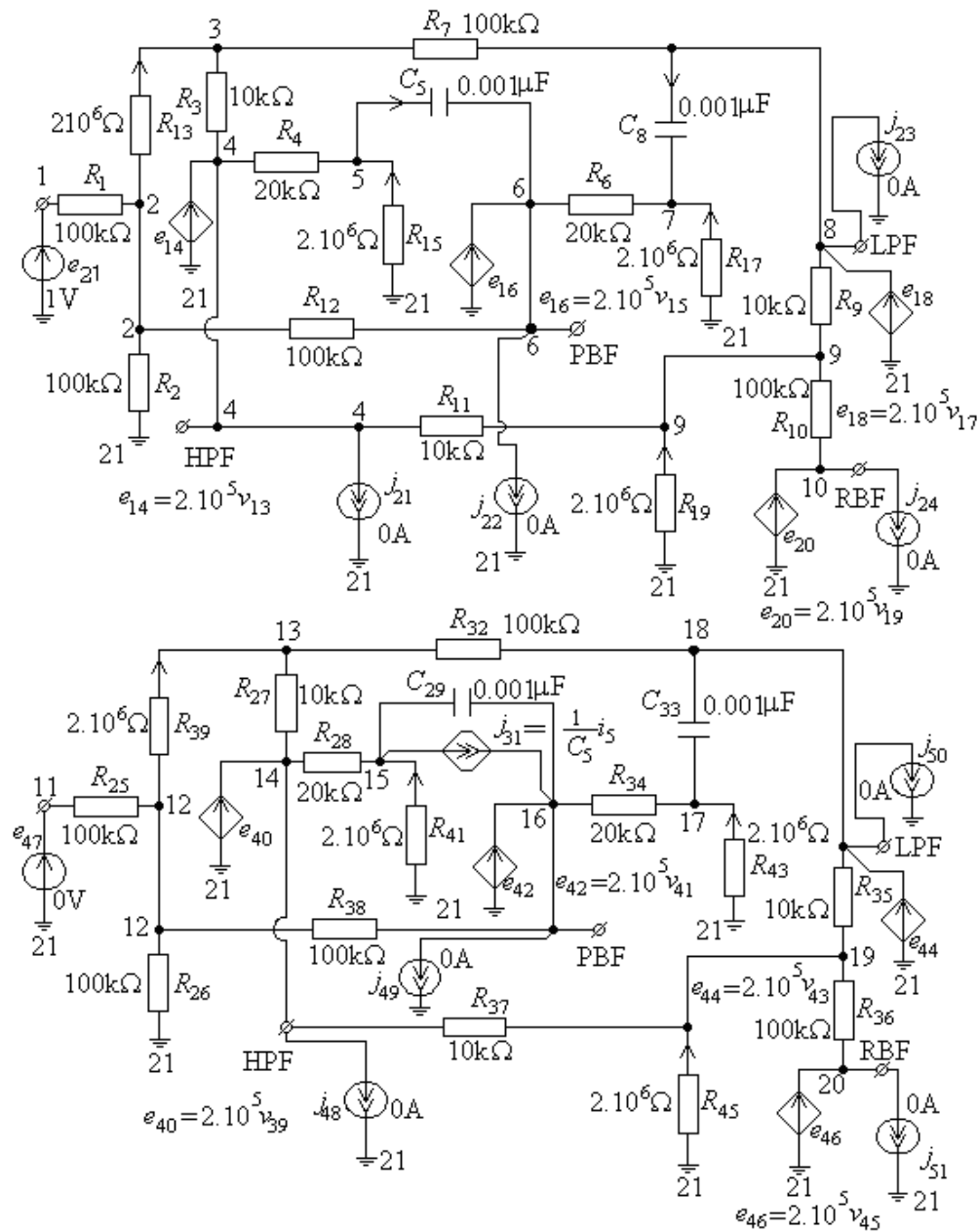


Figure 18: Universal filter.

References

- [1] F. Balik and B. Rodanski, "Simplification of symbolic network functions using large-change sensitivities", in *International Workshop on Symbolic Analysis and Applications in Circuit Design*, 2004, pp. 83-86.
- [2] A. E. Schwarz, *Computer-aided design of microelectronic circuits and systems*. London: Academic Press, 1987.
- [3] A. J. Goldstein and F. F. Kuo, "Multiparameter sensitivity", *IER Transactions CT*, no. 8, pp. 177-178, June 1961.
- [4] L. O. Chua and P. M. Lin, *Computer-aided analysis of electronic circuits*. Englewood Cliffs, N. J.: Prentice Hall, Inc., 1975.
- [5] M. Iordache, Lucia Dumitriu, and I. Matei, "SYAMNM - SYmbolic Analysis based on Modified Nodal Method", User Guide, Electrical Engineering Department, PUB, Bucharest, 1999.

- [6] A. F. Schwarz, "Large-change and differential network sensitivity", *IEEE Transactions on Circuits and Systems*, no. 24, pp. 662-663, Nov. 1977.
- [7] A. F. Schwarz, "Semisymbolic analysis of network sensitivity", *Electronics Letters*, pp. 291-292, April 1978.
- [8] W. R. Perkins and J. B. Cruz, *Engineering of dynamic systems*. New York: Wiley, 1969.
- [9] J. B. Cruz, *System sensitivity analysis*. Stroudsburg: Hutchinson and Ross, 1973.
- [10] S. K. Mitra, *Analysis and synthesis of linear active networks*. New York: Wiley, 1969.
- [11] R. K. Brayton and R. Spence, *Sensitivity and optimization*. Amsterdam: Elsevier, 1980.
- [12] S. W. Director and R. A. Rohrer, "The generalized adjoint network and network sensitivities", *IEEE Transactions*, vol. CT-16, pp. 318-328, Aug. 1969.
- [13] A. Marascu, Lucia Dumitriu, M. Iordache, and D. Niculae, "Tolerance analysis for Chebyshev filters", in *IEEE European Conference on Circuits and Systems for Communications*, 2008, pp. 78-62.
- [14] D. A. Calahan, *Computer-aided network design*. New York: McGraw-Hill, 1972.
- [15] W. J. McCalla, *Fundamentals of computer-aided circuit simulation*. Boston: Kluwer Academic Publishers, 1988.
- [16] M. Iordache and Lucia Dumitriu, "Sensitivity analysis using auxiliary networks", in *International Workshop on Symbolic and Numerical Methods, Modeling and Applications to Circuit Design*, 2006, pp. 23-29.
- [17] M. Iordache and Lucia Dumitriu, "Symbolic state equations for analog circuits", *Revue Roumaine des Sciences Techniques, Série Électrotechnique et Énergétique*, vol 45, no. 1, pp. 21-37, 2000.
- [18] L. Dumitriu and M. Iordache, "A new algorithm for fast generation of network functions in symbolic reduced form", *Revue Roumaine des Sciences Techniques, Série Électrotechnique et Énergétique*, vol. 45, no. 1, pp. 3-19, 2000.

Symbolic Noise Analysis in Analog Circuits

Carlos Sánchez-López*

Autonomous University of Tlaxcala, México and IMSE-CNM, CSIC and University of Sevilla, Spain

Abstract: An approach to the symbolic noise analysis on linear or linearized analog circuits at the transistor level of abstraction is presented. A brief exposition on the signal-path approach into analog circuits working in voltage-mode and current-mode, which are modeled with nullors, is given. Therefore, symbolic noise parameters, such as:

$\overline{V_{n,out}^2}$, $\overline{V_{n,in}^2}$, NF_v , $\overline{I_{n,out}^2}$, $\overline{I_{n,in}^2}$ and NF_i of analog circuits are computed, where all the

noise sources associated to MOS transistors and passive elements are assumed to be uncorrelated. Two examples are introduced to illustrate the potentiality of the approach proposed. The first example is a voltage-mode analog circuit, where the signal-path is approached by using the nullator concept and its properties. On the contrary, in the second example, a current-mode analog circuit is considered where the signal-path is approached by using the norator concept along with its properties.

Keywords: Symbolic nodal analysis, noise, noise figure, nullor, thermal noise, shot noise, flicker noise, power spectral density, amplifiers, current mirrors.

1. Introduction

Nowadays, the need of using portable electronic systems which are operated by batteries, has been a main motivation of that sub-micron technologies to become developed and used to design modern analog circuits. However, as active devices are scaled down, the transistor dimensions are very small, so that second-order effects, such as: noise, distortion, body-effect, channel length modulation among others, before considered of minimal importance, become very important into the design of a modern electronic circuit [1]-[3]. Among all these effects, noise analysis is very important, since it imposes a lower limit of input signal amplitude that can be processed through the analog circuit. Noise analysis can be carried out by using a numeric simulator, such as Hspice [12]. However, it does not show what circuit elements are noisier, since numeric simulators basically serve to verify the circuit performance and a lot of numerical simulations should be executed to understand and predict the small-signal characteristics and the noise behavior.

In this chapter, we will focus on fully-symbolic noise analysis of analog circuits at the transistor level of abstraction. With this choice, we briefly review the noise concept and its classification on the different types of noise, along with the noise sources of resistors and MOS Transistors (MOSTs). Because the behavior of a MOST can be modeled with the nullor, the original analog circuit can be transformed to fully-connected nullor equivalent circuit. Therefore, there are paths allowing the signal to be processed from the input node to the output node. We then approach the Signal-Path (SP) into nullor circuits by applying the nullator and norator properties. Later on, only noise sources of MOSTs and resistors connected to the SP are introduced in the system of equations. In order to compute fully-symbolic noise parameters, the

*Address correspondence to Carlos Sánchez-López: Autonomous University of Tlaxcala, México and IMSE-CNM, CSIC and University of Sevilla, Spain; E-mail: carlsanmx@yahoo.com.mx

formulation method of the system of equations introduced in Chapter 3 is used. Some examples are given to show the effectiveness of the proposed approach.

2. Noise Models

Noise, considered of second importance in the past, becomes significant as technologies are scaled down to deep sub-micron [3]-[6]. Indeed, small dimensions of transistors as well as supply voltages reduction, together with the need of higher performances, make that modern circuits become more sensitive to noise [1]-[6]. Basically, noise is defined as any signal with random amplitude in function of the time. This means that the exact amplitude at any instant of time cannot be predicted, although the past values are known [7],[8]. Further, the noise is interference unrelated to the signal of interest and it can be characterized by a Power Spectral Density (PSD), which shows as the power of a random signal can be carried over one unitary bandwidth, around of a frequency. Noise is caused by the small current and voltage fluctuations, which are generated within passive elements and active devices. Further, the behavior of noise in analog circuits is very important because it represents a lower limit to the size of electrical signal that can be amplified by a circuit without significant deterioration in the signal quality [1],[2],[7]. Although there are several types of noise sources in electrical circuits, here, we discuss only three important noise sources: thermal noise, shot noise and flicker noise.

2.1. Thermal noise

Thermal noise also known as Johnson or Nyquist noise is due to the thermal excitation of charge carriers in a conductor. The Brownian type random motion of electrons produces noise that has a white spectral density, since it is unaffected by the presence or absence of direct current and it is proportional to the absolute temperature [7],[8]. As shown in **Fig. 1**, the thermal noise of a resistor may be modeled as a current source in parallel with the resistor and the mean square value of the noise current source is given by:

$$\overline{I_{n,th}^2} = 4kTG\Delta f \quad (1)$$

where G is the conductance, k is the Boltzman's constant ($1.38 \times 10^{-23} \text{ JK}^{-1}$), T is the absolute temperature in Kelvin and Δf is the noise bandwidth.

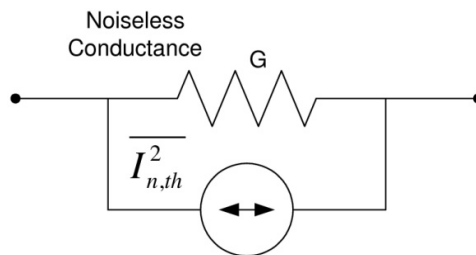


Figure 1: Thermal noise representation in a resistor with a current source.

Also, noise can be modeled by a series voltage source, as shown in **Fig. 2**, where the mean square value is given by:

$$\overline{V_{n,th}^2} = 4kTR\Delta f \quad (2)$$

where R is the resistance.

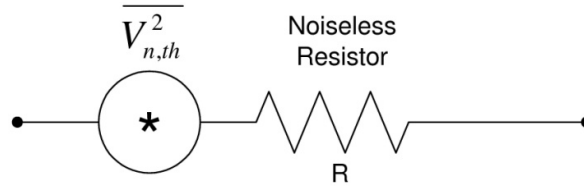


Figure 2: Thermal noise representation in a resistor with a voltage source.

2.2. Flicker noise

Flicker noise or just $1/f$ noise is associated with contact and surface irregularities in semiconductors; hence, it can be found in all the active devices as well as in discrete passive elements [5],[6],[9]-[11]. Flicker noise is caused by contamination and defects in the silicon lattice structure. The PSD of the flicker noise is given by:

$$\frac{\overline{I_{n,1/f}^2}}{\Delta f} = K_1 \frac{I^a}{f^b} \quad (3)$$

where I is a direct current that flows into the device, K_1 is a constant for a particular device, a is a constant in the range: 0.2 to 2, and b is a constant of value around unity.

2.3. Shot noise

Shot noise normally occurs when there is a potential barrier. The P-N junction diode is an example. Shot noise is produced when the electrons and holes randomly cross the potential barrier [7]-[10]. The PSD of the shot noise is given by:

$$\frac{\overline{I_{n,shot}^2}}{\Delta f} = 2qI_{DC} \quad (4)$$

where $q=1.6 \times 10^{-19}$ C is the electron charge and I_{DC} is the direct current. Shot noise is constant as a function of the frequency and it does not change with the temperature. Thermal and shot noise sources are called white noise, because they have a constant magnitude of power over the frequency [7]-[10].

2.4. Noise sources in MOS transistors

An analog integrated circuit is always composed by active devices and passive elements. In particular, resistors, Bipolar Junction Transistors (BJTs) and MOSTs are all noisy devices and each of them has several noise sources associated. For the case of a MOST, it is modeled with four noise current sources [4],[9],[10],[12]. Two of them represent the thermal noise associated with the parasitic drain and source series resistances. The other two are modeled as current sources from drain to source, which

represents the thermal noise and flicker noise, respectively. All the noise sources are characterized in strong inversion by their PSD [9],[10]. The contribution of shot noise in MOSTs is very small and it becomes more important when MOSTs are working in weak inversion, whereas the thermal and flicker noise are present in all operating regions, and the flicker noise source becomes negligible at high frequencies [11]. The noise models for resistors, junction diodes, BJTs and MOSTs are shown in **Fig. 3**. In the same manner, **Fig. 4** shows the noise model of MOSTs implemented in the numeric simulator Hspice, according to the level of simulation and technology dependence [12]. It is worth mentioning that either noise voltage or current sources have not an exact polarity because their value will be squared anyway.



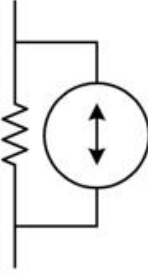


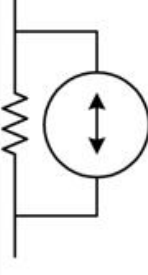

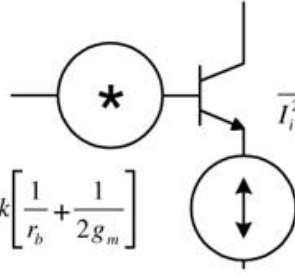
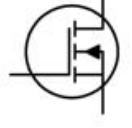
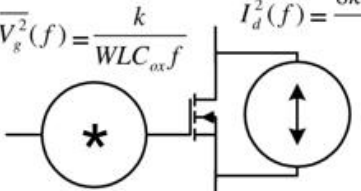
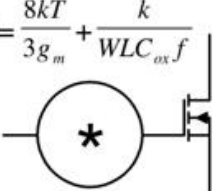
Element	Noise Models	
	 $\overline{V_R^2}(f) = 4kTR$	 $\overline{I_R^2}(f) = \frac{4kT}{R}$
	 $r_D = \frac{kT}{qI_D}$ $\overline{V_R^2}(f) = 2kTr_D$	 $r_D = \frac{kT}{qI_D}$ $\overline{I_R^2}(f) = 2qI_D$
	 $\overline{V_i^2}(f) = 4k \left[\frac{1}{r_b} + \frac{1}{2g_m} \right]$ $\overline{I_i^2}(f) = 2q \left[I_B + \frac{kI_B}{f} + \frac{I_c}{ B(f) ^2} \right]$	
	 $\overline{V_g^2}(f) = \frac{k}{WLC_{ox}f}$ $\overline{I_d^2}(f) = \frac{8kTg_m}{3}$	 $\overline{V_i^2}(f) = \frac{8kT}{3g_m} + \frac{k}{WLC_{ox}f}$

Figure 3: Element noise models.

Hspice Models	Flicker Noise	Thermal Noise
NLEV = 0	$S_{ID} = \frac{K_F I_{IDS}^{AF}}{C_{ox} L_{eff}^2 f}$	$S_{channel} = \frac{8kTg_m}{3}$
NLEV = 1	$S_{ID} = \frac{K_F I_{IDS}^{AF}}{C_{ox} W_{eff} L_{eff} f}$	(Also for NLEV = 2)
NLEV = 2,3	$S_{ID} = \frac{K_F g_m^2}{C_{ox} W_{eff} L_{eff} f^{AF}}$	$S_{channel} = \frac{8kT}{3} B(v_{gs} - v_{TH}) \frac{1+a+a^2}{1+a} GDSNOI$ $a = 1 - \frac{v_{ds}}{v_{dsat}}$ Linear $a = 0$ Saturation
BSIM3v3	http://www-device.eecs.berkeley.edu/~bsim3/get.html	

Figure 4: Noise models in Hspice for MOSTs.

3. Equivalent Input Noise Sources

From the circuit theory point of view, an electric network can be treated as a noisy linear two-port network or a noisy linear four-terminal network for small-signal applications [8]. Fig. 5(a) represents a noisy linear voltage-mode circuit modeled as a two-port network. All the noise sources in an electric circuit can be replaced by an equivalent input-referred noise voltage source $\overline{V_{n,in}^2}$ and an equivalent input-referred noise current source $\overline{I_{n,in}^2}$, as shown in Fig. 5(b) [1],[2],[7],[8].

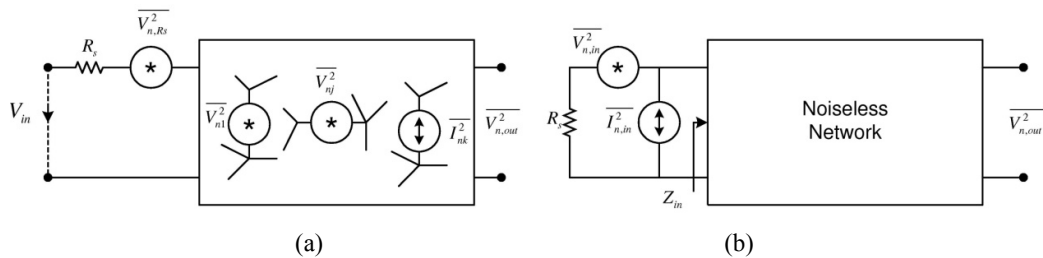


Figure 5: (a) Noisy linear or linearized circuit (b) input-referred noise sources.

To compute the total output noise density given by $\overline{V_{n,out}^2}$, the input in Fig. 5(a) must be short-circuited, taking into account the noise source of the source resistor. All the noise sources are considered as uncorrelated; therefore, the total noise contribution due to each noise source should be computed at the output port by applying the superposition principle and given by:

$$\overline{V_{n,out}^2} = \sum_{j=1}^k \overline{Q_{n,j}^2} TF_{j,out}^2 \tag{5}$$

where $\overline{Q_{n,j}^2}$ is the j -th noise voltage or current source and $TF_{j,out}$ is the transfer function from this noise source to the output of the circuit. In order to compute $\overline{V_{n,in}^2}$, (5) has to be divided by the square of the transfer function. Henceforth, for circuits working in voltage-mode, the input-referred noise voltage source is given as:

$$\overline{V_{n,in}^2} = \frac{\overline{V_{n,out}^2}}{A_v^2} \quad (6)$$

where A_v is the voltage gain. On the other hand, to compute the input-referred noise current source, the input port must be open, thus, the total output noise is given in terms of $\overline{I_{n,in}^2}$, which flows through the input impedance Z_{in} :

$$\overline{V_{n,out}^2} = \overline{I_{n,in}^2} Z_{in}^2 A_v^2 \quad (7)$$

By combining (6) and (7), it follows that:

$$\overline{I_{n,in}^2} = \frac{\overline{V_{n,in}^2}}{Z_{in}^2} \quad (8)$$

The approximation of (6) and (8) are underlying the assumption that the analog circuit is driving a finite source resistance R_s , and finite input impedance. Note that if $R_s=0$, $\overline{I_{n,in}^2}$ flows through $\overline{V_{n,in}^2}$ and has not effect at the output, hence, the noise contribution is solely from $\overline{V_{n,in}^2}$, but if $R_s=\infty$, $\overline{V_{n,in}^2}$ has not effect and the output noise is due only to $\overline{I_{n,in}^2}$. On the other hand, for finite values of R_s , both input-referred noise sources contribute to the equivalent input noise of the circuit. Also, $\overline{V_{n,in}^2}$ and $\overline{I_{n,in}^2}$ are strongly correlated, because they are dependent of the same noise sources. However, in many practical circuits, the correlation is small and may be neglected [7]. As a consequence, for voltage-mode circuits, often R_s is low and Z_{in} is high, hence, the main input noise source is given by (6).

The methodology described above can also be applied to current-mode circuits. Thus, to compute $\overline{I_{n,out}^2}$ with reference to **Fig. 6(a)**, the input-port must be an open-circuit and the output-port must be a short-circuit, as depicted in **Fig. 6(a)**, where the noise contribution due to the source resistance is again considered. Therefore, the total output noise density is given by:

$$\overline{I_{n,out}^2} = \sum_{j=1}^k \overline{Q_{n,j}^2} TF_{j,out}^2 \quad (9)$$

The input-referred noise current source is given by:

$$\overline{I_{n,in}^2} = \frac{\overline{I_{n,out}^2}}{A_i^2} \quad (10)$$

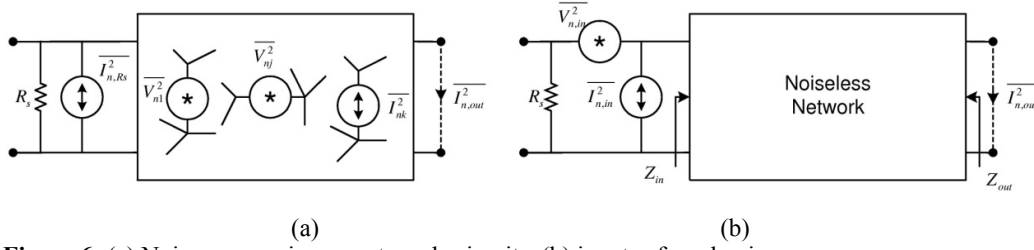


Figure 6: (a) Noise sources in current-mode circuits, (b) input-referred noise sources.

where A_i is the current gain. The input-referred noise voltage source can be obtained by short circuiting the input in order to find the output noise. Here, Z_{in} and $V_{n,in}^2$ are in series and a Norton equivalent circuit can be used for these two-terminal element. Therefore, $I_{n,out}^2$ can be approximated to:

$$I_{n,out}^2 = \frac{V_{n,in}^2}{Z_{in}^2} A_i^2 \quad (11)$$

From (10) and (11), it follows that:

$$V_{n,in}^2 = I_{n,in}^2 Z_{in}^2 \quad (12)$$

Both equations given by (10) and (12) are valid for any finite source resistance and finite input impedance, as mentioned above for voltage-mode circuits and shown in **Fig. 6(b)**. The input impedance levels at current-mode circuits are often designed to achieve low impedance levels, whereas the impedance level of the source resistance is always high. Therefore, the main input noise source is given by (10). Further, if the total output noise voltage is computed instead of the total output noise current, $I_{n,out}^2$ can be computed by considering (5) and the output impedance Z_{out} :

$$I_{n,out}^2 = \frac{V_{n,out}^2}{Z_{out}^2} \quad (13)$$

All the equations, from (5) to (13) are very useful to compute noise parameters related with the design of electronic circuits. Furthermore, they can be used within a fully-symbolic noise analysis method in order to compute the total noise density and the input-referred noise sources of analog circuits.

4. Noise Factor and Noise Figure

The output total noise densities along with the gain of an electronic system are the most important factors in order to describe the noise figure. Noise Figure (NF) is a measure of degradation of the Signal to Noise Ratio (SNR), caused by all the components in the SP of electronic circuits. The most commonly accepted definition for NF is given by:

$$NF = \frac{SNR_{in}}{SNR_{out}} \Big|_{T=300^{\circ}K} \quad (14)$$

with SNR_{in} and SNR_{out} defined as the SNR measured at the input and output terminals, respectively [7]. Alternatively, NF can be defined in terms of dB units and given by:

$$NF_{db} = 10 \log \left(\frac{SNR_{in}}{SNR_{out}} \right) \Big|_{T=300^{\circ}K} \quad (15)$$

Note, however, that (14) is sometimes referred as *noise factor*, in order to distinguish (15) from (14). Here, the NF term is used throughout the chapter. Also, SNR is a term for the average power ratio between a signal and noise, therefore, the signal and noise powers must be measured at the same terminals of a circuit and within the same bandwidth, as shown in **Fig. 7**. Furthermore, the noise and the signal are often measured across the same impedance. As a consequence, the SNR can be computed by using square amplitude ratio and (14) can be rearranged to obtain the NF equation for circuits working in voltage-mode. A similar analysis is carried out for current-mode circuits, where the NF equation becomes:

$$NF_v = \frac{\overline{V_{n,out}^2}}{A_v^2 \overline{V_{n,R_s}^2}}, \quad NF_i = \frac{\overline{I_{n,out}^2}}{A_i^2 \overline{V_{n,R_s}^2}} \quad (16)$$

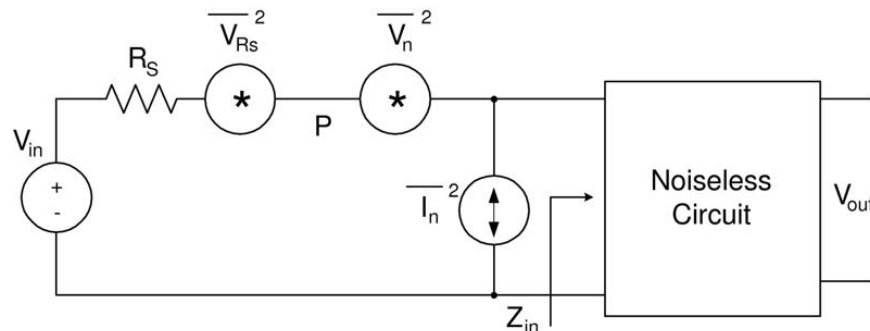


Figure 7: Approximation of the NF in two-port networks.

5. Nullor-Based MOST Noise Model

The behavior of BJTs and MOSTs are modeled traditionally with a voltage-controlled current source. On the other hand, controlled sources can also be modeled with nullators and norators, as shown in **Fig. 5** of Chapter 3. In this manner, by using this nullor-based model and applying the nullor properties, the ideal behavior of BJTs or MOSTs can be achieved. In the same manner as in Chapter 3, not only parasitic elements but also noise sources can easily be added, as shown in **Fig. 8** and **9**, respectively [13].

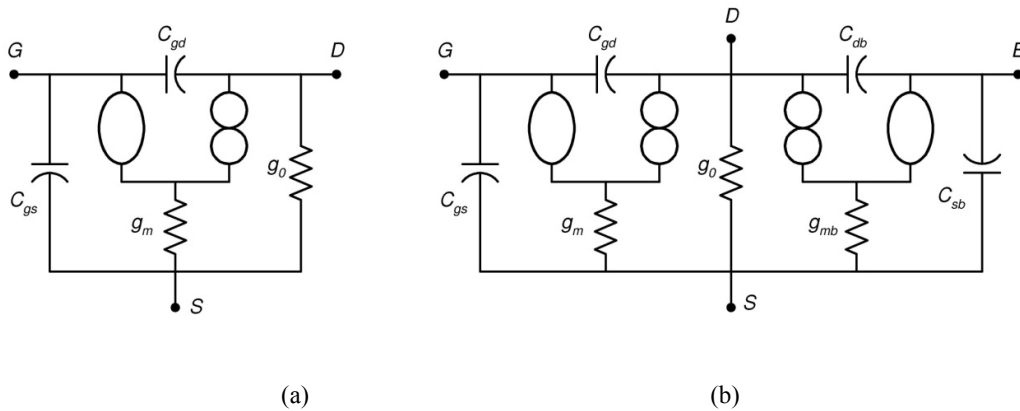


Figure 8: Nullor-based model of a MOST as: (a) three-terminal active device, (b) four-terminal active device.

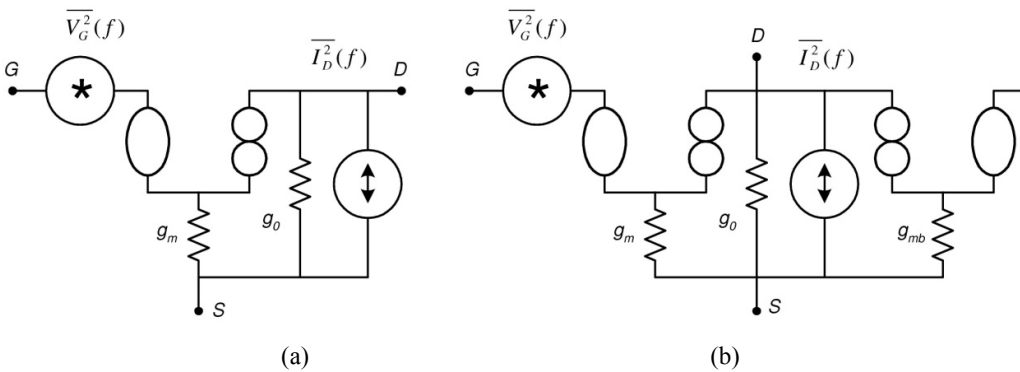


Figure 9: Nullor-based noise model of a MOST as: (a) three-terminal active device (b) four-terminal active device.

6. Signal-Path Approximation in Analog Circuits

In the previous sections, we briefly review the noise concept, its classification and some types of noise presents in active and passive elements. Also, the equations of *NF* for voltage- and current-mode circuits along with the nullor-based models of MOSTs were presented. In this section, the SP approximation in analog circuits is discussed in order to find the most important noise sources, which contributes with higher noise levels in the total output noise.

Symbolic analysis techniques have been proposed in the literature in order to obtain fully-symbolic small-signal characteristics. However, it is well known that for large circuits, the generated symbolic equation grows exponentially with the size of the circuit. In response to this problem, simplification techniques have been introduced in order to approximate the dominant symbolic terms, see for instance, Chapter 7. On the other hand, analog circuits at the transistor level of abstraction can be transformed to nullor equivalent circuits by using the nullor-based model of MOSTs shown in **Fig. 8**. As a consequence, the equivalent circuit is fully-connected, so that there are paths allowing the signal to be processed from the input to the output terminal. Indeed, SP is defined as a trajectory or trajectories where the signal travels through a circuit, from the input node to the output node but non-touching bias or reference nodes. A SP-node is one through which the signal passes.

6.1. Signal-path in voltage-mode circuits

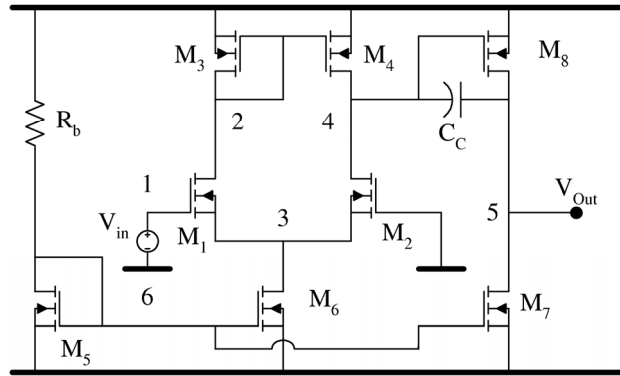
From the equivalences of nullators and admittances given in Chapter 3, the voltage across an admittance connected between two nullators is the same as if the admittance were connected alone. In this manner, the SP can be approached by the following steps [14]:

1. Model each MOST and signal voltage source with their nullor-based models, as shown in **Fig. 9** (the nullor-based model of the signal voltage source is given in Chapter 3). Besides, do not include the noise sources and the output conductance of a MOST connected as diode.
2. Find composed pairs of nullator-transconductance having the following structure: $[O_x g_{mx}, node-, node-, node+]$. Here, the first term is the label of the pair, the second term is the negative node of O_x , the third term is the negative node of the transconductance and the fourth term is the positive node of the norator associated, which should be momentarily eliminated.
3. The pair of nodes of each two-terminal element must be compared with the input node. If the input node is equal to any node of the element, the other node of the element is the new node to compare.
4. Eliminate all two-terminal elements that are not connected to the SP.
5. If the first node of a composed pair is equal to the reference node and by applying the rule (c) from **Fig. 5**, in Chapter 3, the second node of the composed pair is a virtual ground. Therefore, all elements connected to this node are eliminated along with the node, since the voltage across in the nullator is zero.
6. Save the number of nodes into the SP.
7. Since the nullor is composed by a nullator and norator pair, the norator elements will be included with their corresponding nullator.

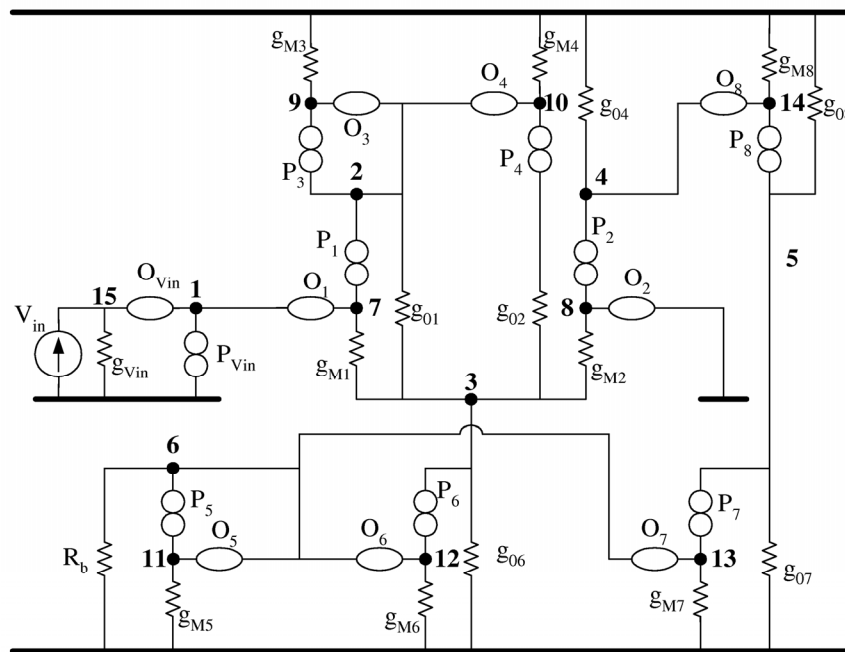
As an example, let us consider the analog circuit shown in **Fig. 10(a)**. The SP can be approached following the steps described above. Thus, the nullor equivalent circuit is shown in **Fig. 10(b)**. Later on, the transconductance and nullator element of each MOST are joined in order to obtain nullator-transconductance pairs, as shown in **Fig. 11(a)**. The step 3 is executed on this simplified circuit; therefore, the SP-nodes are obtained as: 1, 2, 3, 4, 5, 7, 8, 9, 10, 14 and 15. All two-terminal elements that are not connected to any SP-node should be deleted. Now, by applying the step 5, one can see that the node 3 is a virtual ground, since it is a node with low impedance and should be eliminated. The number of SP-nodes is 9 and finally, the reduced circuit is shown in **Fig. 11(b)**, where the nullator elements have been added.

6.2. Signal-path in current-mode circuits

For the case of current-mode circuits, the norator properties should be used in order to approximate the SP [15]. According to the norator definition, the current that enters and leaves its positive and negative nodes is the same. As a consequence any arbitrary amount of current can be supplied to an element connected between two norators. Basically, the SP approach in current-mode circuits is a modification of the SP approach in voltage-mode circuits, which can be explained in the following steps:



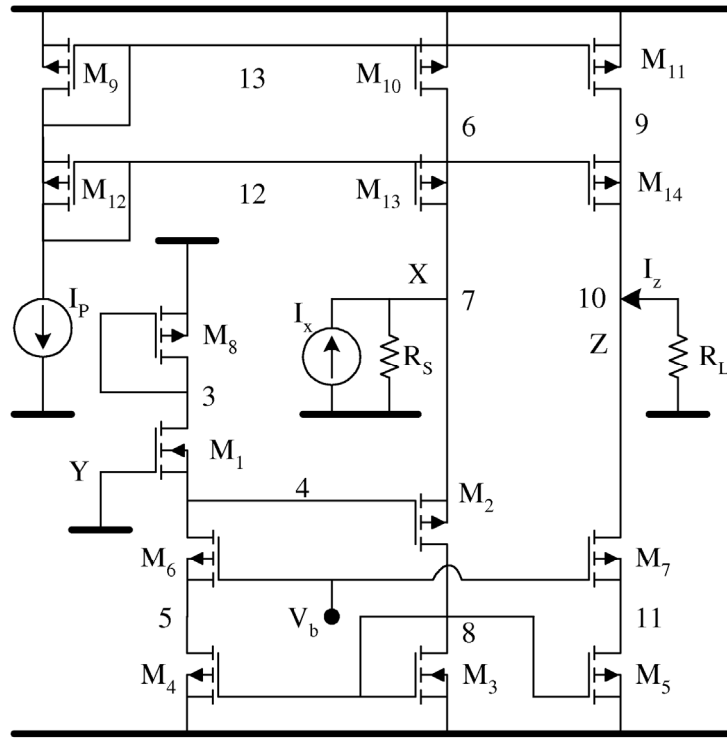
(a)



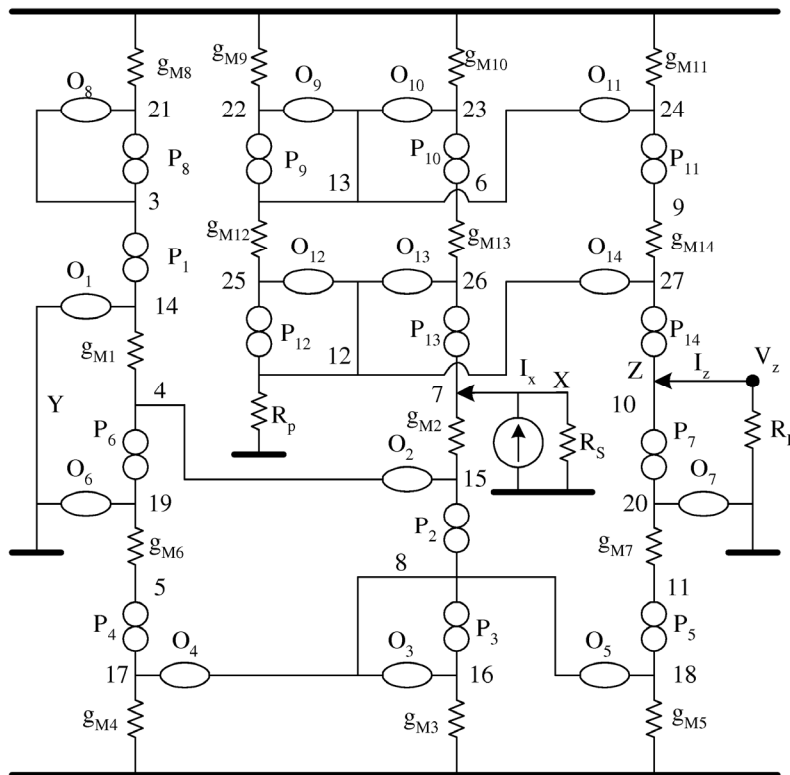
(b)

Figure 10: (a) OTA-Miller circuit (b) nullor equivalent circuit.

1. Model each MOST with its nullor-based model, as shown in Fig. 9. It is well known that the maximum current flows by those elements with low impedance and since the numerical value of g_{mx} of a MOST is higher than the numerical value of g_{0x} , then does not include the output conductance.
2. Find composed pairs of norator-transconductance elements. For this new pair, the associated nodes to norators, nullators and transconductances are included as follows: $[P_x g_{mx}, node-, node-, node+]$. The first term is the label of the pair, the second term is the positive node of P_x , the third term is the negative node of the transconductance and finally, the negative node O_x . Here, nullator elements should be momentarily eliminated since the current through it is zero.
3. Starting from the input node, compare it with the pair of nodes of all two-terminal elements. If the input node is equal to any node of the pair of nodes



(a)



(b)

Figure 12: (a) CCII+ (b) nullor equivalent circuit.

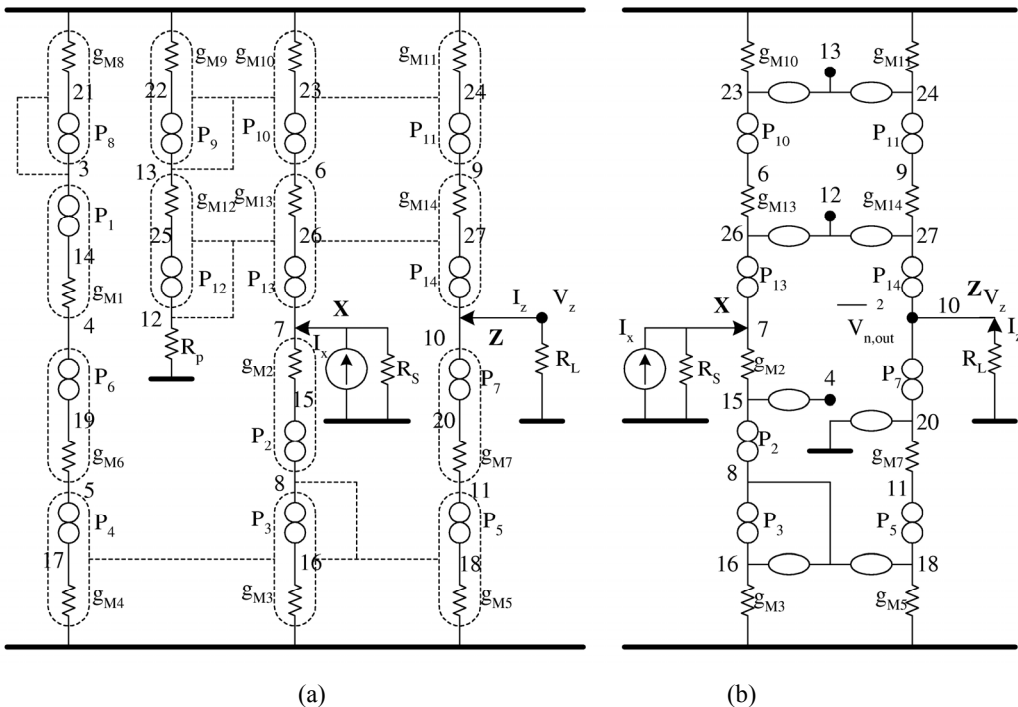


Figure 13: (a) Simplified nullor equivalent circuit (b) reduced nullor circuit.

7. Symbolic Noise Analysis Method

Before introducing the symbolic noise analysis method, we show the noise analysis of the common-source MOS feedback amplifier circuit taken from [1], which is shown in Fig. 14 along with its noise sources. For this example, the ideal behavior of I_p is assumed; therefore, its noise contribution is insignificant.

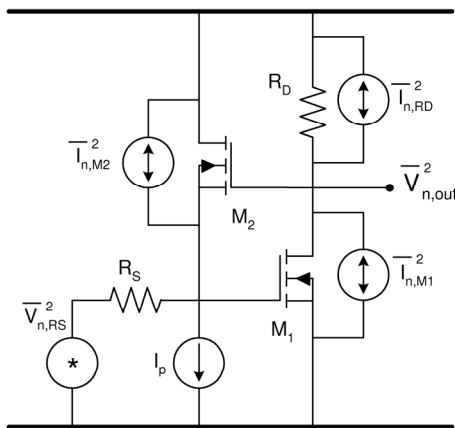


Figure 14: Feedback amplifier with noise sources.

According to (5), the individual contribution from each noise source to the output should be computed. The output noise source due to the noise voltage of the source resistance is given as:

$$\overline{V_{n,out}^2} = \overline{V_{n,R_S}^2} A_v^2 \tag{17}$$

The noise current of M_2 flows through of R_s , generating output noise given by:

$$\overline{V_{n,out2}^2} = \overline{I_{n,M_2}^2} R_s^2 A_v^2 \quad (18)$$

The noise current due to R_D and M_1 is multiplied by the output resistance and given by:

$$\overline{V_{n,out3}^2} = \left(\overline{I_{n,M_2}^2} + \overline{I_{n,R_D}^2} \right) R_{out}^2 \quad (19)$$

The voltage gain A_v , and the output resistance R_{out} , are computed by using the nullor-based models shown in **Fig 8** and the formulation method of the system of equations introduced in Chapter 3:

$$A_v = \frac{-g_{M1} R_D}{1 + g_{M2} R_s (1 + g_{M1} R_D)}, \quad R_{out} = \frac{(1 + g_{M2} R_s) R_D}{1 + g_{M2} R_s (1 + g_{M1} R_D)} \quad (20)$$

Because noise sources are assumed to be uncorrelated, every term in (20) must be squared. Therefore, by using (5), the total output noise density is given by:

$$\overline{V_{n,out}^2} = \frac{g_{M1}^2 R_D^2 (\overline{V_{n,R_s}^2} + \overline{I_{n,M_2}^2} R_s^2) + (\overline{I_{n,M_1}^2} + \overline{I_{n,R_D}^2}) (1 + g_{M2}^2 R_s^2) R_D^2}{1 + g_{M2}^2 R_s^2 (1 + g_{M1}^2 R_D^2)} \quad (21)$$

7.1. Noise analysis in voltage-mode circuits

One of the attractive aspects on the use of nullors is that not only nullor-based models but also noise current sources are used to model the behavior of MOSTs. Therefore, the total output noise can also be computed by using algebraic symbolic analysis methods. The proposed noise analysis method can be summarized in the following steps:

1. Find the SP by applying the method given in subsection 6.1.
2. Include the noise current source of each resistor and MOST. The signal voltage source must be short-circuited.
3. Formulate the system of equations by using the proposed method in Chapter 3, but now all transconductances and resistors should be squared.

Let us consider one more time the analog circuit shown in **Fig. 14**. The noise analysis is as follows:

Step 1. The nullor equivalent circuit from **Fig. 14** is shown in **Fig. 15**. Easily, it can be observed that the SP includes the following elements: R_s , O_1 , g_{M1} , O_2 , g_{M2} , R_D , along with SP-nodes given by: 1, 2, 3, 4 and the signal voltage source has been short-circuited, as shown in **Fig. 15**. Besides, for this single example, the output conductance associated to M_1 and M_2 are not taken into account.

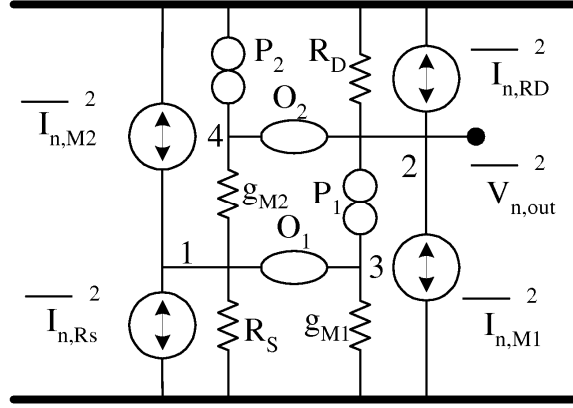


Figure 15: Nullor equivalent circuit from Fig. 14.

Step 2. The incorporation of the noise sources of resistors and MOSTs is shown in Fig. 15.

Step 3. The formulation of the system of equations is given by:

$$\begin{bmatrix} \overline{I_{n,M2}^2} + \overline{I_{n,Rs}^2} \\ \overline{I_{n,M1}^2} + \overline{I_{n,RD}^2} \end{bmatrix} = \begin{bmatrix} g_{M2}^2 + \frac{1}{R_S^2} & -g_{M2}^2 \\ g_{M1}^2 & \frac{1}{R_D^2} \end{bmatrix} \begin{bmatrix} \overline{V_{1,3}^2} \\ \overline{V_{2,4}^2} \end{bmatrix} \quad (22)$$

Therefore, the total noise output density is obtained to $\overline{V_{n,out}^2} = \overline{V_{2,4}^2}$ and given as:

$$\overline{V_{n,out}^2} = \frac{g_{M1}^2 R_S^2 R_D^2 (\overline{I_{n,Rs}^2} + \overline{I_{n,M2}^2}) + (\overline{I_{n,M1}^2} + \overline{I_{n,RD}^2}) (1 + g_{M2}^2 R_S^2) R_D^2}{1 + g_{M2}^2 R_S^2 + g_{M1}^2 g_{M2}^2 R_S^2 R_D^2} \quad (23)$$

Comparing (21) with (23) one can see that both equations represent the same total output noise density. On the other hand, to compute the input-referred noise sources and NF_v , (16), (20) and (23) should be applied.

$$\overline{V_{n,in}^2} = \frac{1 + g_{M2}^2 R_S^2}{g_{M1}^2} (\overline{I_{n,M1}^2} + \overline{I_{n,RD}^2}) + R_S^2 (\overline{I_{n,Rs}^2} + \overline{I_{n,M2}^2}) \quad (24)$$

$$\overline{I_{n,in}^2} = g_{M2}^2 (1 + g_{M1}^2 R_D^2) \left[\frac{1 + g_{M2}^2 R_S^2}{g_{M1}^2} (\overline{I_{n,M1}^2} + \overline{I_{n,RD}^2}) + R_S^2 (\overline{I_{n,Rs}^2} + \overline{I_{n,M2}^2}) \right] \quad (25)$$

where

$$Z_{in}^2 = \frac{1}{g_{M1}^2 g_{M2}^2 R_D^2 + g_{M2}^2} \quad (26)$$

$$NF_v = \frac{1 + g_{M2}^2 R_S^2}{g_{M1}^2} \left(\frac{\overline{I_{n,M1}^2} + \overline{I_{n,RD}^2}}{\overline{V_{n,Rs}^2}} \right) + R_S^2 \left(\frac{\overline{I_{n,Rs}^2} + \overline{I_{n,M2}^2}}{\overline{V_{n,Rs}^2}} \right) \quad (27)$$

Considering the noise model for each element shown in Fig. 3 and 4 with NLEV=0, the fully-symbolic expression of NF_v is given by:

$$NF_v = 1 + \frac{2}{3} g_{M2} R_S + \left(\frac{1}{R_D} + \frac{2}{3} g_{M1} \right) \frac{1 + g_{M2}^2 R_S^2}{g_{M1}^2 R_S} \quad (28)$$

Here, only the contribution of thermal noise is presented and the equation including both flicker and thermal noise is given in [17]. On the other hand, the symbolic noise analysis of the OTA Miller circuit shown in Fig. 10(a) is given in [18], where all the noise sources into the circuit have been considered.

7.2. Noise analysis in current-mode circuits

In the same manner as in subsection 7.1, the total output noise to current-mode circuits can also be computed by using algebraic symbolic analysis methods. The proposed noise analysis method can be summarized in the following steps [19]:

1. Find the SP by applying the method given in subsection 6.2.
2. Include the noise current source of each resistor and MOST. The signal current source must be an open-circuit.
3. Formulate the system of equations by using the proposed method in Chapter 3, with all magnitudes of transconductances and resistors squared.

For this case, we use the circuit shown in Fig. 13(b). Following the described above method, the noise sources associated to resistors and MOST are included in the reduced nullor circuit, as shown in Fig. 16.

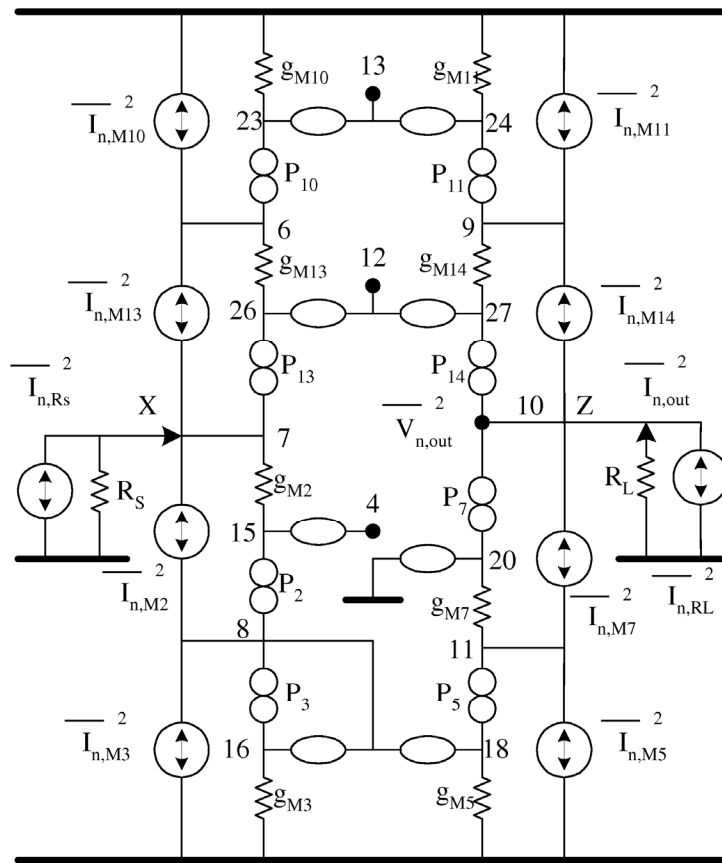


Figure 16: Reduced nullor circuit including noise sources.

After, formulating the system of equations by applying the formulation method introduced in Chapter 3, it is obtained as:

$$\begin{bmatrix} \overline{I_{n,M10}^2} + \overline{I_{n,M13}^2} \\ \overline{I_{n,M2}^2} + \overline{I_{n,M13}^2} + \overline{I_{n,Rs}^2} \\ \overline{I_{n,M2}^2} + \overline{I_{n,M3}^2} \\ \overline{I_{n,M11}^2} + \overline{I_{n,M14}^2} \\ \overline{I_{n,M7}^2} + \overline{I_{n,M14}^2} + \overline{I_{n,RL}^2} \\ \overline{I_{n,M2}^2} + \overline{I_{n,M7}^2} \end{bmatrix} = \begin{bmatrix} g_{M13}^2 & 0 & 0 & 0 & 0 & 0 \\ -g_{M13}^2 & g_{M2}^2 + g_S^2 & 0 & 0 & 0 & 0 \\ 0 & -g_{M2}^2 & g_{M3}^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & g_{M14}^2 & 0 & 0 \\ 0 & 0 & 0 & -g_{M14}^2 & -g_L^2 & -g_{M7}^2 \\ 0 & 0 & g_{M5}^2 & 0 & 0 & g_{M7}^2 \end{bmatrix} \begin{bmatrix} \overline{V_6^2} \\ \overline{V_7^2} \\ \overline{V_{8,16,8}^2} \\ \overline{V_9^2} \\ \overline{V_{10}^2} \\ \overline{V_{11}^2} \end{bmatrix} \quad (29)$$

The solution of (29) is carried out by applying Cramer's rule to $\overline{V_{n,out}^2} = \overline{V_{10}^2}$ and $\overline{I_{n,out}^2} = \overline{V_{n,out}^2} R_L^{-2}$, where the following equalities are assumed:

$$\begin{aligned} g_{M5}^2 &= g_{M4}^2 = g_{M3}^2, & g_{M14}^2 &= g_{M13}^2 = g_{M11}^2 = g_{M10}^2 = g_{M9}^2 \\ \overline{I_{n,M14}^2} &= \overline{I_{n,M13}^2} = \overline{I_{n,M11}^2} = \overline{I_{n,M10}^2} = \overline{I_{n,M9}^2}, & \overline{I_{n,M5}^2} &= \overline{I_{n,M4}^2} = \overline{I_{n,M3}^2} \end{aligned} \quad (30)$$

Thus, the total output noise density is given as:

$$\begin{aligned} \overline{V_{n,out}^2} &= \frac{4R_L^2 g_{M2}^2 R_S^2 \overline{I_{n,M7}^2}}{1 + g_{M2}^2 R_S^2} + \frac{6R_L^2 g_{M2}^2 R_S^2 \overline{I_{n,M9}^2}}{1 + g_{M2}^2 R_S^2} + \frac{R_L^2 g_{M2}^2 R_S^2 \overline{I_{n,RL}^2}}{1 + g_{M2}^2 R_S^2} + \frac{R_L^2 \overline{I_{n,M2}^2}}{1 + g_{M2}^2 R_S^2} + \frac{2R_L^2 g_{M2}^2 R_S^2 \overline{I_{n,M3}^2}}{1 + g_{M2}^2 R_S^2} \\ &+ \frac{R_L^2 g_{M2}^2 R_S^2 \overline{I_{n,Rs}^2}}{1 + g_{M2}^2 R_S^2} + \frac{4R_L^2 g_{M2}^2 R_S^2 \overline{I_{n,M2}^2}}{1 + g_{M2}^2 R_S^2} + \frac{4R_L^2 \overline{I_{n,M7}^2}}{1 + g_{M2}^2 R_S^2} + \frac{R_L^2 \overline{I_{n,M9}^2}}{1 + g_{M2}^2 R_S^2} + \frac{R_L^2 \overline{I_{n,RL}^2}}{1 + g_{M2}^2 R_S^2} + \frac{2R_L^2 \overline{I_{n,M3}^2}}{1 + g_{M2}^2 R_S^2} \end{aligned} \quad (31)$$

$$\begin{aligned} \overline{I_{n,out}^2} &= \frac{4g_{M2}^2 R_S^2 \overline{I_{n,M7}^2}}{1 + g_{M2}^2 R_S^2} + \frac{6g_{M2}^2 R_S^2 \overline{I_{n,M9}^2}}{1 + g_{M2}^2 R_S^2} + \frac{g_{M2}^2 R_S^2 \overline{I_{n,RL}^2}}{1 + g_{M2}^2 R_S^2} + \frac{\overline{I_{n,M2}^2}}{1 + g_{M2}^2 R_S^2} + \frac{2g_{M2}^2 R_S^2 \overline{I_{n,M3}^2}}{1 + g_{M2}^2 R_S^2} \\ &+ \frac{g_{M2}^2 R_S^2 \overline{I_{n,Rs}^2}}{1 + g_{M2}^2 R_S^2} + \frac{4g_{M2}^2 R_S^2 \overline{I_{n,M2}^2}}{1 + g_{M2}^2 R_S^2} + \frac{4\overline{I_{n,M7}^2}}{1 + g_{M2}^2 R_S^2} + \frac{\overline{I_{n,M9}^2}}{1 + g_{M2}^2 R_S^2} + \frac{\overline{I_{n,RL}^2}}{1 + g_{M2}^2 R_S^2} + \frac{2\overline{I_{n,M3}^2}}{1 + g_{M2}^2 R_S^2} \end{aligned} \quad (32)$$

In order to compute the input-referred noise sources and NF_i , (16) and (32) should be used.

$$\begin{aligned} \overline{I_{n,in}^2} &= R_S^2 \overline{I_{n,Rs}^2} + 4R_S^2 \overline{I_{n,M7}^2} + 6R_S^2 \overline{I_{n,M9}^2} + R_S^2 \overline{I_{n,RL}^2} + 2R_S^2 \overline{I_{n,M3}^2} + 4R_S^2 \overline{I_{n,M2}^2} + \frac{4\overline{I_{n,M7}^2}}{g_{M2}^2} + \frac{\overline{I_{n,M9}^2}}{g_{M2}^2} \\ &+ \frac{\overline{I_{n,RL}^2}}{g_{M2}^2} + \frac{2\overline{I_{n,M3}^2}}{g_{M2}^2} + \frac{2\overline{I_{n,M2}^2}}{g_{M2}^2} \end{aligned} \quad (33)$$

$$\overline{V_{n,in}^2} = \frac{\overline{I_{n,in}^2}}{g_{M2}^2}, \quad Z_{in} = \frac{1}{g_{M2}^2} \quad (34)$$

$$\begin{aligned}
NF_i = 1 &+ \frac{4\overline{I_{n,M7}^2}}{\overline{I_{n,Rs}^2}} + \frac{6\overline{I_{n,M9}^2}}{\overline{I_{n,Rs}^2}} + \frac{\overline{I_{n,RL}^2}}{\overline{I_{n,Rs}^2}} + \frac{2\overline{I_{n,M3}^2}}{\overline{I_{n,Rs}^2}} + \frac{4\overline{I_{n,M2}^2}}{\overline{I_{n,Rs}^2}} + \frac{4\overline{I_{n,M7}^2}}{g_{M2}^2 R_S^2 \overline{I_{n,Rs}^2}} + \frac{\overline{I_{n,M9}^2}}{g_{M2}^2 R_S^2 \overline{I_{n,Rs}^2}} \\
&+ \frac{\overline{I_{n,RL}^2}}{g_{M2}^2 R_S^2 \overline{I_{n,Rs}^2}} + \frac{2\overline{I_{n,M3}^2}}{g_{M2}^2 R_S^2 \overline{I_{n,Rs}^2}} + \frac{\overline{I_{n,M2}^2}}{g_{M2}^2 R_S^2 \overline{I_{n,Rs}^2}}
\end{aligned} \quad (35)$$

where the current gain is given by:

$$A_i = \frac{g_{M2} g_{M5} R_S}{g_{M3} (1 + g_{M2} R_S)} \quad (36)$$

Finally, if only the contribution of thermal noise is considered, then (35) can be simplified as:

$$\begin{aligned}
NF_i = 1 &+ \frac{8R_S g_{M7}}{3} + 4R_S g_{M9} + \frac{R_S}{R_L} + \frac{4R_S g_{M3}}{3} + \frac{8R_S g_{M2}}{3} + \frac{8g_{M7}}{3g_{M2}^2 R_S} + \frac{2g_{M9}}{3g_{M2}^2 R_S} \\
&+ \frac{1}{R_L g_{M2}^2 R_S} + \frac{4g_{M3}}{3g_{M2}^2 R_S} + \frac{2}{3g_{M2}^2 R_S}
\end{aligned} \quad (37)$$

A comparison between (31) and Hspice is shown in **Fig. 17**, where the contribution of flicker noise has been included. It is worthy mentioning that the symbolic noise parameters have been computed in function of noise current sources, which are associated to resistors and MOSTs. Therefore, these noise current sources can be substitute by their noise models depending of the operation region of MOSTs and technology used [20]. Furthermore, once that input-referred noise sources have been computed, the following step is to accomplish a symbolic noise analysis at the circuit level of abstraction, as shown in [21].

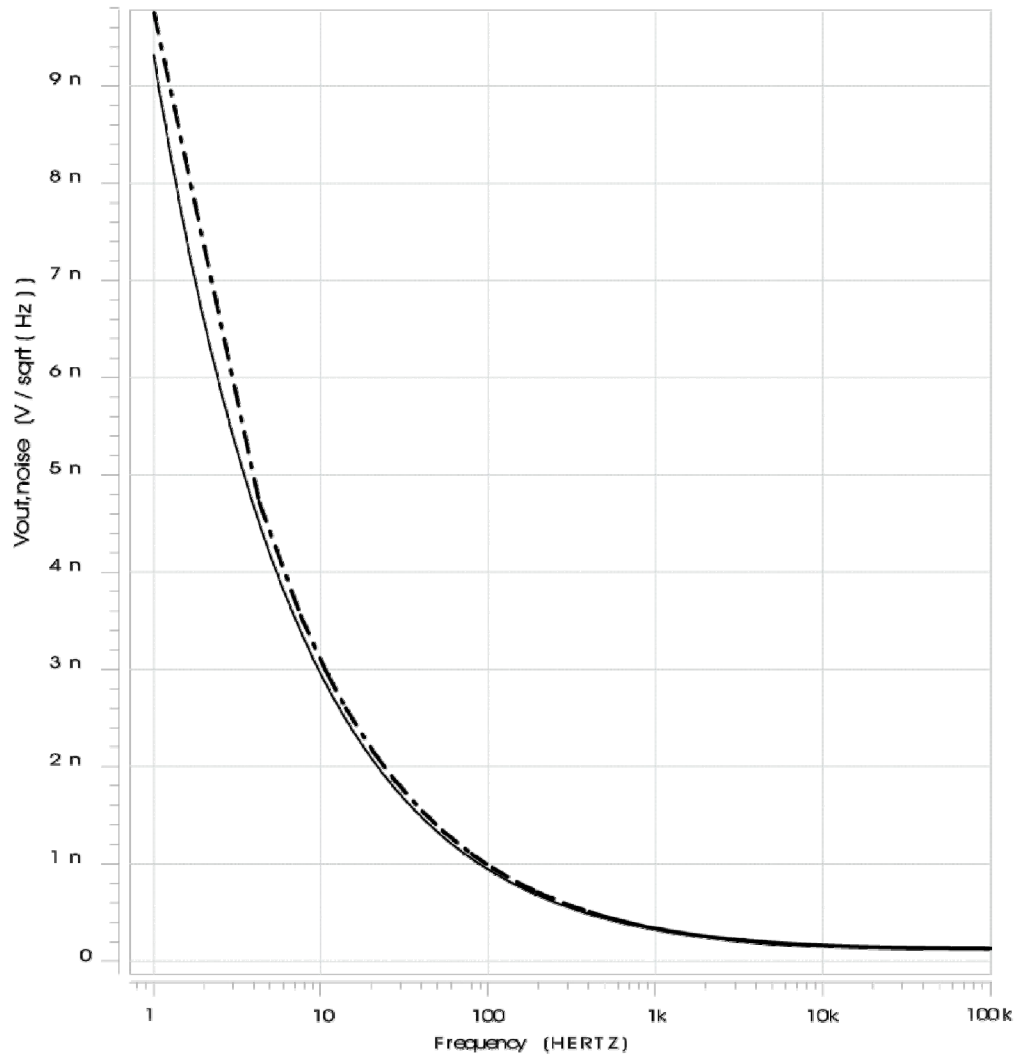


Figure 17: Numerical comparisons (Hspice solid-line, Symbolic dashed-line).

8. Conclusion

In this chapter, we basically present two fully-symbolic noise analysis methods of analog circuits, not only for voltage-mode circuits but also for current-mode circuits. As one can see throughout the chapter, the symbolic noise analysis methods are based on the SP approximation of analog circuits at the transistor level of abstraction. Further, the nullor-based model of a MOST as a three- or four-terminal active device has been introduced, including parasitic elements and noise sources. To avoid the high computation costs associated with the formulation of the system equations, the formulation method described in the Chapter 3 has been used. As a consequence, it was demonstrated that the total output noise density, the input-referred noise sources and noise figure can quickly be computed. Finally, we discuss some examples in order to show the usefulness of the proposed methods.

Acknowledgements

The preparation of this chapter has been partially supported by Promep-Mexico under the project number: UATLX-PTC-088 and by Consejería de Innovación, Ciencia y

Empresa, Junta de Andalucía under the project number: TIC-2532. The author thanks the support of the JAE-Doc program of CSIC, co-funded by FSE.

References

- [1] B. Razavi, *RF microelectronics*. NJ: Prentice-Hall, 1998.
- [2] B. Razavi, *Design of analog CMOS integrated circuits*. USA: McGraw-Hill, 2001.
- [3] Q. Huang, F. Piazza, P. Orsatti, and T. Ohguro, "The impact of scaling down to deep submicron on CMOS RF circuits", *IEEE Journal of Solid State Circuits*, vol. 33, pp. 1023-1036, July 1998.
- [4] A. Dastgheib, B. Murmann, "Calculation of total integrated noise in analog circuits", *IEEE Transactions on Circuits and Systems I: Regular papers*, vol. 55, pp. 2988-2993, November 2008.
- [5] A. Arnaud, C. Galup-Montoro, "A compact model for flicker noise in MOS transistors for analog circuit design", *IEEE Transactions on Electron Devices*, vol. 50, pp. 1815-1818, August 2003.
- [6] T. Noulis, S. Siskos, G. Sarraayrouse, "Analysis and selection criteria of BSIM4 flicker noise simulation models", *International Journal of Circuit Theory and Applications*, vol. 14, pp. 813-823, 2008.
- [7] A. Van Der-Ziel, *Noise, characterization, measurement*, Prentice-Hall, 1970.
- [8] M. S. Gupta, *Selected papers on noise in circuits and systems*, IEEE Press, USA: NY, 1987.
- [9] R. P. Jindal, "Compact noise models for MOSFETs", *IEEE Transactions on Electron Devices*, vol. 53, pp. 2051-2061, September 2006.
- [10] A. Arnaud, C. Galup-Montoro, "Consistent noise models for analysis and design of CMOS circuits", *IEEE Transactions on Circuits and Systems I*, vol. 51, pp. 1909-1915, October 2004.
- [11] C. H. Chen and M. J. Deen, "High frequency noise of MOSFETs I: Modeling", *Elsevier Solid-State Electronics*, vol. 42, pp. 2069-2081, May 1998.
- [12] Hspice User Guide: *Simulation and analysis*, version B-2008.09, September 2008.
- [13] E. Tlelo-Cuautle, C. Sánchez-López, "Symbolic computation of NF of transistor circuits", *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E87-A, pp. 2420-2425, September 2004.
- [14] C. Sánchez-López and E. Tlelo-Cuautle, "Behavioral model generation for symbolic analysis of analog integrated circuits", in *IEEE International Symposium on Signals, Circuits and Systems*, 2005, pp. 327-330.
- [15] C. Sánchez-López and E. Tlelo-Cuautle, "Behavioral model generation of current-mode analog circuits", in *IEEE International Symposium on Circuits and Systems*, 2009, pp. 2761-2764.
- [16] C. Hyeong-Woo, O. Satomi, and W. Kenzo, "Class A CMOS Current Conveyors", *IEICE Transactions on Fundamentals of Electronics*, vol. E81-A, pp. 1164-1167, June 1998.
- [17] C. Sánchez-López, E. Tlelo-Cuautle, A. Díaz-Sánchez, "Computing the noise figure of MOST circuits by applying symbolic analysis", in *IEEE Proceeding Midwest Symposium on Circuits and Systems*, 2002, pp. 429-432.
- [18] C. Sánchez-López and E. Tlelo-Cuautle, "Symbolic Noise Analysis in Analog Integrated Circuits", in *IEEE International Symposium on Circuits and Systems*, 2004, pp. 245-248.
- [19] C. Sánchez-López, E. Tlelo-Cuautle, M. Fakhfakh, M. Loulou, "Computing simplified noise-symbolic-expressions in CMOS CCs by applying SPA and SGA", in *IEEE International Conference on Microelectronics*, 2007, pp. 147-150.
- [20] T. Heng-Fa, J. Sheng-Lyang, and M. H. Juang, "A unified model for high-frequency current noise of MOSFETs", *Elsevier Solid-State Electronics*, vol. 47, pp. 2043-2048, May 2003.
- [21] C. Sánchez-López and E. Tlelo-Cuautle, "Symbolic noise analysis in Gm-C filters", in *IEEE Conference of Electronics, Robotics and Automotive Mechanics*, 2006, pp. 49-53.

CHAPTER 11

Symbolic Pole/Zero Analysis

Francisco V. Fernández^{1,*}, Carlos Sánchez-López², Rafael Castro-López³ and Elisenda Roca-Moreno³

¹Dept. Electronics and Electromagnetism, University of Sevilla and IMSE-CNM, CSIC; ²IMSE-CNM, CSIC and University of Sevilla, Spain, and ³IMSE-CNM, CSIC and University of Sevilla

Abstract: Extraction of pole/zero expressions as a function of circuit parameters has traditionally been an essential tool for designers. In this Chapter, the main specific techniques for symbolic pole/zero extraction are described and their pros and cons are discussed. The application of the different techniques is illustrated with experimental results on practical circuits.

Keywords: Poles and zeros, symbolic pole-zero extraction, root splitting, error control, simplification after generation, generalized eigenvalue problem, Haley's modification-decomposition, standard eigenvalue problem, time-constant matrix, simplification before generation, simplification during generation, QR algorithm, QZ algorithm.

1. Introduction

Symbolic analysis tools automate the analysis, usually of linear (or linearized) circuits, in which all or part of the circuit parameters are kept in symbolic form. The output is usually in the form of a network function in the complex frequency variable:

$$H(s) = \frac{\sum_{j=1}^m s^j f_j(x)}{\sum_{i=1}^n s^i g_i(x)} \quad (1)$$

In addition to the network functions, *e.g.* magnitude or phase of the voltage gain of an amplifier, poles and zeros provide extremely valuable design knowledge. A direct extraction of symbolic poles and zeros from previously calculated network functions like (1) usually yields poor results. The first reason is that only second-order polynomials can be considered. Although analytical solutions for third-order and fourth-order polynomials exist, they are impractical in a symbolic analysis context. On the other hand, as Chapter 7 illustrates, the network function complexity grows exponentially with the circuit size. Therefore, the calculation of symbolic poles and zeros inherits this problem; and it even worsens as a number of operations between network function coefficients has to be performed.

Therefore, the application of approximated extraction techniques together with the application of approximation techniques to the generation of symbolic expressions becomes mandatory. In this chapter, main reported techniques are discussed. In Section 2, the classical root splitting technique for approximated root extraction is

*Address correspondence to Francisco V. Fernández: Dept. Electronics and Electromagnetism, University of Sevilla and IMSE-CNM, CSIC; E-mail: pacov@imse-cnm.csic.es

introduced. This technique is combined together with simplification after generation techniques for reduction of expression complexity. As the plain application of simplification after generation techniques still limits dramatically the circuit size that can be analyzed, extraction techniques that make use of simplification before generation techniques (approximation of the network equations before they are symbolically solved) are discussed in Section 3. A more mature technique is introduced in Section 4, that uses the time-constant approach, enabling the introduction of simplification before and during generation techniques¹.

2. Classical Root Splitting Technique

Analytical calculation of poles and zeros from the symbolic numerator and denominator polynomials in (1) is theoretically limited to fourth-order polynomials, but practical use in the symbolic analysis context is limited to second-order polynomials. This implies that symbolic pole-zero extraction would be impossible except for the simplest circuits. To avoid this, common approximations performed in manual analysis like the root splitting technique were the first techniques applied for symbolic pole/zero extraction.

If we assume that the n roots of the denominator in (1) are real, then the denominator can be written as:

$$D(s) = g_0 \left(1 - \frac{s}{p_1}\right) \left(1 - \frac{s}{p_2}\right) \cdots \left(1 - \frac{s}{p_n}\right) \quad (2)$$

From the comparison with (1), it follows that:

$$\begin{aligned} g_1 &= g_0 \sum_{i=1}^n \left(-\frac{1}{p_i}\right) \\ g_2 &= g_0 \sum_{i=1}^n \sum_{j=i}^n \frac{1}{p_i p_j} \\ g_3 &= g_0 \sum_{i=1}^n \sum_{j=i}^n \sum_{k=j}^n \left(-\frac{1}{p_i p_j p_k}\right) \\ &\quad \dots \end{aligned} \quad (3)$$

Under the assumption that the first pole is at a much lower frequency than the rest of poles:

$$\|p_1\| \ll \|p_2\|, \dots, \|p_n\| \quad (4)$$

the first equation in (3) can be approximated as:

¹ See Chapter 7 for a detailed discussion of simplification before, during and after generation techniques.

$$g_1 \approx g_0 \left(-\frac{1}{p_1} \right) \quad (5)$$

and therefore:

$$p_1 \approx -\frac{g_0}{g_1} \quad (6)$$

Under the assumption that the second pole is at a much lower frequency than higher order poles, the same procedure can also be applied resulting in:

$$p_2 \approx -\frac{g_1}{g_2} \quad (7)$$

The same procedure can be applied iteratively for higher order poles. A similar procedure can also be applied if two complex poles (or two close real poles) are at a sufficiently lower frequency than other poles:

$$\|p_1\| \approx \|p_2\| \quad \text{and} \quad \|p_1\| \ll \|p_3\|, \dots, \|p_n\| \quad (8)$$

Then, the pair of real or complex conjugate poles can be extracted from a second-order polynomial:

$$D(s) \approx g_0 + g_1 s + g_2 s^2 \quad (9)$$

An analogous mechanism can be followed for the zeros of (1).

This technique was first implemented in [1, 2]. Since the complexity of symbolic network functions grows exponentially with the circuit size, the root splitting technique is not able to produce symbolic expressions of reasonable size for practical circuits by itself. Therefore, it must be applied on approximated functions, obtained from the application of *Simplification After Generation* (SAG) techniques². In these techniques, the exact symbolic network function is first generated and, afterwards, the least significant terms are pruned while some error criterion is met. For instance, given any network function coefficient in (1), formulated as a sum-of-product of symbolic parameters:

$$h_k(\mathbf{x}) = \sum_{l=1}^L h_{kl}(\mathbf{x}) \quad (10)$$

the approaches in [2, 3] eliminate the P least significant terms while the following condition is satisfied:

² Refer to Chapter 7 for a detailed description of these techniques.

$$\frac{\sum_{l=1}^P |h_{kl}(\mathbf{x}_o)|}{\sum_{l=1}^L |h_{kl}(\mathbf{x}_o)|} \leq \varepsilon \quad (11)$$

where ε is a user-defined accuracy margin and \mathbf{x}_o is the point of the design parameter space selected for the evaluation of the approximation error.

If the poles and zeros are to be extracted from the simplified network function, and in order to keep the root positions unchanged, the same error threshold ε can be applied for each coefficient of the network function. However, this may lead to important errors in pole and zero locations due to the very nature of the simplification process. This process consists in eliminating individual symbolic terms of each coefficient and, therefore, the estimated coefficient value varies at discrete steps. Then, the real error in each polynomial coefficient will be different and this may yield very significant shifts of root locations. To avoid this problem, the approach in [1] eliminates the least significant terms in each coefficient incrementally. A small value of ε is used initially and it is increased by $\Delta\varepsilon$ in each iteration. At each iteration, poles and zeros are monitored. If the root shifts are larger than a predefined value, symbolic approximation is stopped. A possible problem is that roots cannot be easily traced when discrete approximations are applied, *e.g.* elimination of certain terms may yield unexpected root shifts with difficult correlation with the previous location.

As an example of application let us consider the cascode amplifier in **Fig. 1(a)** where the small-signal model in **Fig. 1(b)** is used for the MOS transistors³. The application of the technique in [1] provides symbolic expressions for the first two poles and the first zero:

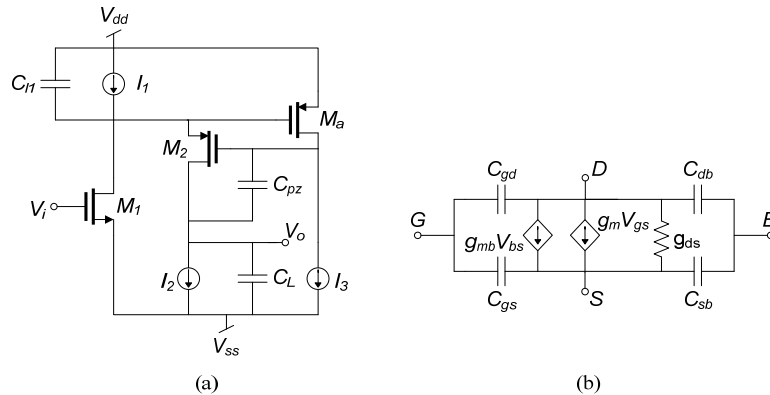


Figure 1: (a) Active feedforward compensated cascode amplifier and (b) small-signal MOS transistor model.

³ The experimental results shown in this chapter are intended for illustrative purposes and coarse comparison. Accurate comparison is infeasible as the implementations of most approaches are not freely or commercially available and we are restricted to the results available in the literature. In many cases, the different approaches have been tested on different circuits. Even in those cases that have been tested on the same circuit, the results may not be comparable as they may be influenced by the extraction technique itself, by the nominal value of the symbolic parameters (around which the simplification is performed) and by the configuration settings of each approach. In most cases, a complete information has not been disclosed.

$$\begin{aligned}
p_1 &= -\frac{g_{ds1}g_{ds2}g_{dsa}}{g_{m2}g_{ma}(C_L + C_{gd2} + C_{pz})} \\
p_2 &= -\frac{g_{ma}(C_L + C_{gd2} + C_{pz})}{(C_{gd2} + C_{pz})(C_L + C_{db1} + C_{gd1} + C_{l1})} \\
z_1 &= -\frac{g_{ma}}{(C_{gd2} + C_{pz})}
\end{aligned} \tag{12}$$

Valuable information can be extracted from (12), like the value of the capacitor C_{pz} that cancels the pole-zero pair:

$$C_{pz} = C_{db1} + C_{gd1} + C_{l1} - C_{gd2} \tag{13}$$

A posterior technique tried to reduce the error in the pole-zero locations by applying one or more Newton-Raphson iterations on the locations predicted by the root splitting technique [4]. A pole/zero at the n th iteration of the Newton-Raphson algorithm is given by:

$$s_{n+1} = s_n - \frac{F(s_n)}{F'(s_n)} \tag{14}$$

This involves the symbolic computation of the quotient of a polynomial (numerator or denominator of the network function evaluated at the symbolic solution of the root obtained in the previous iteration) and its derivative with respect to the complex frequency variable s . In general, the price to pay is a significant increase of the complexity of the symbolic expressions, making the interpretability of the symbolic results more difficult.

The major problems of the root splitting technique applied to symbolic pole/zero extraction are:

- a) The extraction technique shares the limitations of simplification after generation techniques. Remind from Chapter 7 that simplification after generation techniques require the previous generation of the exact network function. The exponential growth of expression complexity with the circuit size limits the maximum analyzable circuit size to a few tens of elements (circuit elements after semiconductor devices have been replaced by their small-signal models).
- b) The approximation may yield significant pole/zero errors. Techniques that avoid this, like [1], may lead to significant increase of expression complexity as the simplification may be stopped prematurely.

3. SBG-Based Techniques

Another group of techniques are based on the application of Simplification Before Generation (SBG) techniques on a matrix formulation of the network equations. For generic linear circuits the Modified Nodal Analysis formulation offers a suitable

procedure to build a set of linear equations in the Laplace domain. Besides, this method allows a clear splitting between the static and dynamic components of the circuit, that is,

$$Y_{MNA} = \mathbf{G} + s\mathbf{C} \quad (15)$$

where the matrix \mathbf{G} contains the static elements (*e.g.*, conductances, controlled sources) and the matrix \mathbf{C} contains the dynamic elements (capacitors and inductors). To ease the subsequent formulation and implementation, inductors are considered in impedance form [5].

Poles are defined as those complex values that lead to the following set of homogeneous equations:

$$(\mathbf{G} + s\mathbf{C})\mathbf{x} = \mathbf{0} \quad (16)$$

or equivalently:

$$\mathbf{G}\mathbf{x} = -s\mathbf{C}\mathbf{x} \quad (17)$$

Solution of (17) corresponds to the well-known generalized eigenvalue problem. The usual mechanism for the numerical solution of (17) is the QZ algorithm [6]. This algorithm can be used for the numerical calculation of poles and zeros but its iterative nature prevents its use for symbolic pole/zero extraction.

To overcome the circuit size limitations of symbolic pole/zero extraction methodologies based on simplification after generation techniques, new approaches have been reported in [7]-[9] whose distinctive feature is the introduction of simplification before generation techniques.

The approach in [7] is based on a simplification before generation technique reported in [10]. The key step is the elimination of device parameters from the cofactors of the nodal matrix if the error induced is below a given error margin. The error is evaluated at a few samples (usually three points per decade) of the frequency range of interest. The frequency range has to be selected to isolate a small cluster of poles or zeros, resembling the classical root splitting technique. Then, the determinant dimension is tried to be reduced by factoring out rows and columns with a single symbol or non-zero entry and performing row and column operations that reduce the number of symbols or non-zero entries. Determinant expansion is performed trying to keep a factorized form that can ease extraction of poles and zeros although it is not clear how the factorization can be performed systematically to yield the appropriate form for pole/zero extraction.

The application of this technique to the Miller op amp in **Fig. 2** in the frequency range from dc to 1KHz allows to obtain the following expression for the first pole [7]:

$$p_1 = -\frac{(g_{ds2} + g_{ds4})(g_{ds6} + g_{ds7})}{g_{m6}C_c} \quad (18)$$

The subsequent application to the frequency range which goes from 7MHz to 70MHz provides an easy factorization of three symbolic poles [7]:

$$\begin{aligned}
 p_2 &= -\frac{C_c g_{m6} + (g_{ds6} + g_{ds7})(C_{gb6} + C_{gd2} + C_{gd4} + C_{gs6})}{(C_c + C_{gd7})(C_{gb6} + C_{gd2} + C_{gd4} + C_{gs6})} \\
 p_3 &= -\frac{(g_{m1} + g_{m2})}{(C_{gb1} + C_{gb2} + C_{gs1} + C_{gs2})} \\
 p_4 &= -\frac{g_{m3}}{(C_{gb3} + C_{gb4} + C_{gd1} + C_{gd4} + C_{gs3} + C_{gs4})}
 \end{aligned} \tag{19}$$

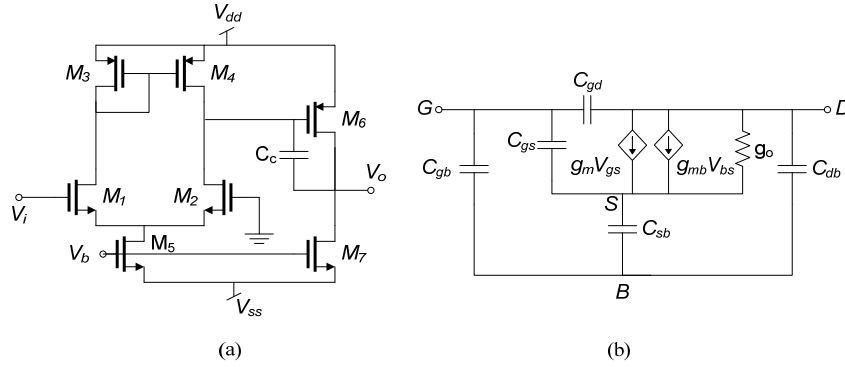


Figure 2: (a) Two-stage CMOS Miller opamp and (b) small-signal MOS transistor model.

However, pole p_3 is not a valid pole as it cancels with an analogous zero at the numerator [7]. This is probably due to the fact that the evaluation of the errors introduced by device parameter elimination is performed in the corresponding cofactor and not in the complete network function.

On the other hand, there is no clear procedure to select the frequency range and the number of samples that provides the right expression for easier root factorization. Small root shifts may produce very significant magnitude/phase errors depending on the root location in the complex frequency plane, therefore stopping the simplification process much before than needed, hence, yielding very complex symbolic expressions. On the contrary, important root displacements may keep undetected by magnitude/phase error criteria if the number of samples is not appropriate. The problem is palliated by increasing the number of samples at the price of additional computation time.

The approach in [8] also performs a simplification before generation technique and tries to avoid the problems of simplification techniques based on magnitude/phase error control. In this case, a single root is selected a priori and the errors induced by parameter eliminations are not evaluated in the cofactors but in the root itself. This could be calculated by the application of the QZ algorithm. However, this algorithm presents two drawbacks in this application. First, although we are interested in a single root, the algorithm calculates the complete root spectrum. And second, the computational cost of the algorithm is relatively high, making inefficient its application for each device parameter elimination that is tried. To avoid this problem,

[8] uses the sensitivity of the root to each device parameter, assuming that the sensitivity is directly related to the root shift caused by cancelling that device parameter. This becomes a weakness of this approach as a small local sensitivity is not a guarantee for a small large-change sensitivity represented by the device parameter elimination.

Once all device parameters with limited impact on pole/zero locations have been eliminated, the determinant is expanded and a polynomial in the complex frequency is obtained. Simplification-after-generation techniques are applied in the coefficients of this polynomial. Hopefully, the polynomial degree is low enough for symbolic extraction of the root. Again additional simplification after generation is tried in the resulting symbolic expression.

Another approach based on simplification before generation techniques to extract poles and zeros was reported in [9]. As the previous one it is based on the approximation of the matrices in (16) for a selected eigenvalue by ranking the eigenvalue shifts induced by different device parameter eliminations and performing the least significant device prunings while some error criterion in the eigenvalue shift is met. The eigenvalue shift is obtained from a linear prediction formula derived from a Taylor series approximation of the generalized eigenvalue problem, similar to the sensitivity analysis above, yielding a ranking of candidate parameter eliminations.

To solve the problem of underestimation of real errors due to significantly bigger large-change sensitivity, this approach estimates pole-zero shifts for each parameter elimination that is performed. For the same reasons above, error tracking by the application of the QZ algorithm is not efficient enough. Therefore, an error tracking by iterative eigenvalue improvement process is applied. If the eigenvalue shift induced by a device parameter elimination is small, we can assume that the nominal value of the eigenvalue is a good initial guess for an iterative calculation process that converges to the perturbed eigenvalue. If the method fails to converge, we can assume that the eigenvalue shift is too large.

Like in previous methods that monitor pole/zero shifts there is a risk of a bad identification of an eigenvalue with its perturbed counterpart. To avoid this, a very significant improvement is introduced in [9], where a Modal Assurance Criterion (MAC) is defined, that measures the correlation of two eigenvectors and that is given as [9]:

$$MAC(\mathbf{u}, \mathbf{u}^*) = \frac{|\mathbf{u}^H \mathbf{u}^*|^2}{(\mathbf{u}^H \mathbf{u})(\mathbf{u}^{*H} \mathbf{u}^*)} \quad (20)$$

where \mathbf{u} is the eigenvector corresponding to the selected eigenvalue of the original system in (16) and \mathbf{u}^* is the eigenvector of the perturbed system.

If a perturbed eigenvector corresponds to the original one, they must be closely correlated and therefore, the modal assurance criterion must be close to 1. By the contrary, if the modal assurance criterion is much smaller, it means that the eigenvectors do not correspond and the approximation is not valid, even if they are numerically very close.

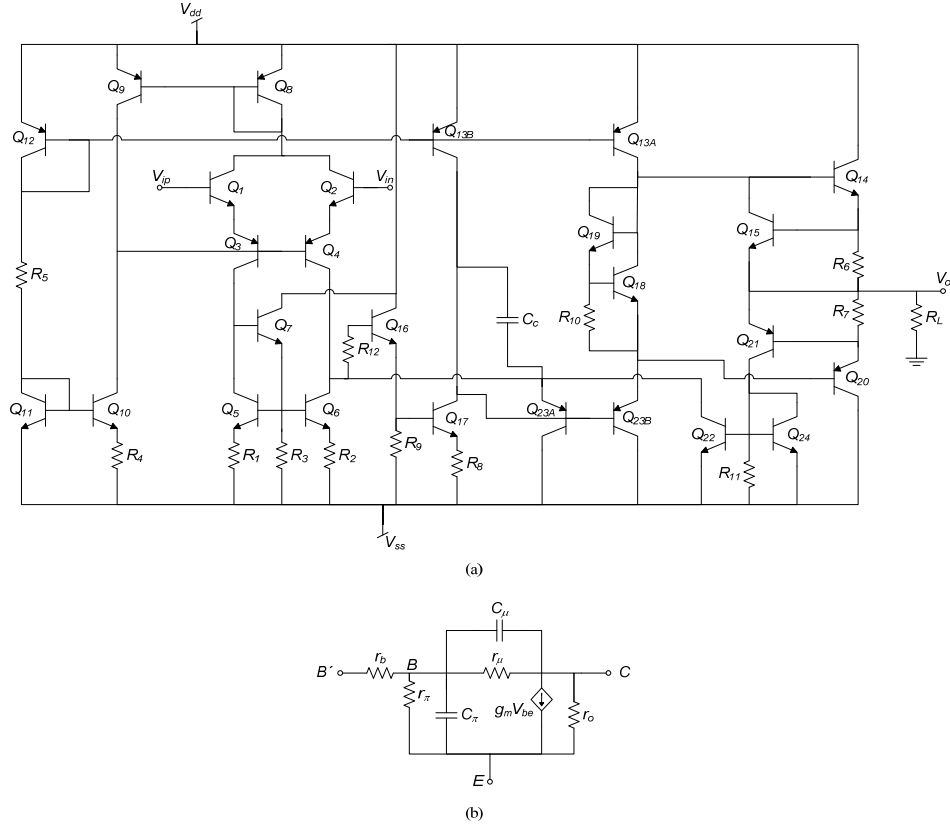


Figure 3: (a) μA741 operational amplifier and (b) small-signal bipolar transistor model.

The determinant of the matrix resulting after the simplification before generation technique is expanded. Hopefully again, a low-order polynomial is obtained from which the desired root can be extracted. Simplification after generation techniques can be applied for further formula simplification. Robust simplification during generation techniques like those reviewed in Chapter 7, are not in general applicable as they cannot be applied on matrix formulations.

As illustrative examples, several experimental results have been reported with this approach, like the first two poles of the uA741 operational amplifier in **Fig. 3**.

By applying simplification before generation techniques based on magnitude/phase error control the first pole is obtained:

$$p_1 = -\frac{(r_{o13B} + r_{o17})(R_9 r_{o4} + g_{m17} R_8 r_{\pi17}(r_{o4} + g_{m16} R_9 r_{\pi16}))}{C_1 g_{m16} g_{m17} R_9 r_{o13B} r_{o17} r_{o4} r_{\pi16} r_{\pi17}} \quad (21)$$

However, the second pole cannot be obtained with this technique due to the existence of a close zero. However, the approach in [9] allows obtaining a symbolic expression for the second pole:

$$p_2 = -\frac{1}{R_{11}(C_{\mu21} + C_{\mu22} + C_{\pi22} + C_{\pi24})} \quad (22)$$

The two-stage amplifier in **Fig. 4** is used as a second example. An error bound of 5% in pole-zero shifts allows to obtain symbolic expressions for the first two poles and the first zero:

$$\begin{aligned}
 p_1 &= -\frac{(g_{ds2} + g_{ds4})(g_{ds6} + g_{ds7})}{g_{m6}C_c + (g_{ds2} + g_{ds4})C_L} \\
 p_2 &= -\frac{g_{ds9}(C_c C_{gs6} + C_c C_L + C_L C_{gs6}) - \sqrt{g_{ds9}^2 (C_c C_{gs6} + C_c C_L + C_L C_{gs6})^2 - 4C_c^2 C_L C_{gs6} g_{ds9} g_{m6}}}{2C_c C_L C_{gs6}} \\
 z_1 &= \frac{g_{ds9} g_{m6}}{C_c (g_{ds9} - g_{m6})}
 \end{aligned} \quad (23)$$

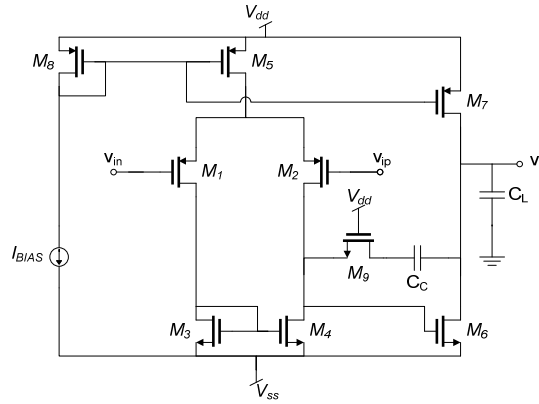


Figure 4: Two-stage Miller amplifier.

4. The Time-Constant Approach

4.1. Haley's modification-decomposition method

To arrive at a solution for the symbolic pole-zero extraction, [11] proposes an approach inspired on Haley's modification-decomposition method [12, 13]. Haley's modification-decomposition method starts by premultiplying (17) by \mathbf{G}^{-1} :

$$\mathbf{x} = -s\mathbf{G}^{-1}\mathbf{C}\mathbf{x} \quad (24)$$

Assuming that the circuit contains M capacitors, matrix \mathbf{C} can be expressed as:

$$\mathbf{C} = \mathfrak{I}\mathbf{C}_d\mathfrak{I}^T \quad (25)$$

where \mathbf{C}_d is the following diagonal matrix:

$$\mathbf{C}_d = \begin{bmatrix} C_1 & 0 & \cdots & 0 \\ 0 & C_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & C_M \end{bmatrix} \quad (26)$$

And the capacitor incidence matrix \mathfrak{S} is given by:

$$\mathfrak{S} = [\mathbf{q}_1 \quad \mathbf{q}_2 \quad \cdots \quad \mathbf{q}_M] \quad (27)$$

where the vector \mathbf{q}_k corresponding to the k -th capacitor connected between nodes i and j :

$$\mathbf{q}_k = [0 \quad \cdots \quad \overset{i}{1} \quad \cdots \quad \overset{j}{-1} \quad \cdots \quad 0]^T \quad (28)$$

By premultiplying (24) by \mathfrak{S}^T and replacing (25) it yields:

$$\mathfrak{S}^T \mathbf{x} = -s \mathfrak{S}^T \mathbf{G}^{-1} \mathfrak{S} \mathbf{C}_d \mathfrak{S}^T \mathbf{x} \quad (29)$$

By redefining its components:

$$\begin{aligned} \mathbf{x}_M &= \mathfrak{S}^T \mathbf{x} \\ \mathbf{R} &= \mathfrak{S}^T \mathbf{G}^{-1} \mathfrak{S} \\ \mathbf{T} &= \mathbf{R} \mathbf{C}_d \end{aligned} \quad (30)$$

Eq. (29) becomes:

$$\mathbf{x}_M = -s \mathbf{T} \mathbf{x}_M \quad (31)$$

By performing the following variable change:

$$\tau = -\frac{1}{s} \quad (32)$$

(31) becomes:

$$(\mathbf{T} - \tau \mathbf{1}) \mathbf{x}_M = \mathbf{0} \quad (33)$$

This is called a modification of matrix \mathbf{G} and its effect is the transformation of an N -dimensional generalized eigenvalue problem into an M -dimensional standard eigenvalue problem, whose eigenvalue spectrum can be related to that of the generalized eigenvalue problem by using (32).

As reported in [12], an important advantage of this transformation is that the numerical calculation of the roots can be performed by using the QR algorithm, faster and more stable than the QZ algorithm.

However, the biggest advantage of this formulation in the symbolic context, as will be shown in the following, is the simple structure of the matrix \mathbf{T} that provides valuable physical insight on the dynamic circuit behavior. Moreover, each entry of the matrix \mathbf{T} can be symbolically calculated very efficiently, being the only approach

that allows introducing both simplification before and during generation techniques, enabling the highest accuracy with the smallest expression complexity.

4.2. The time-constant matrix

Let us consider the structure of matrix \mathbf{T} in (30). The element located in row α and column β is given by:

$$T_{\alpha\beta} = \mathbf{q}_\alpha^T \mathbf{G}^{-1} \mathbf{q}_\beta C_\alpha \quad (34)$$

In this equation we can identify a vector

$$\mathbf{y} = \mathbf{G}^{-1} \mathbf{q}_\beta \quad (35)$$

that represents the nodal voltages when the resistive part of the circuit is excited by a single source placed at the port defined by capacitor C_β . Then, the linear combination

$$\mathbf{q}_\alpha^T \mathbf{y} \quad (36)$$

represents the voltage across the port defined by capacitor C_α . Therefore, the matrix element can be interpreted as the transresistance from port β to port α multiplied by capacitor C_α and matrix \mathbf{T} becomes:

$$\mathbf{T} = \begin{bmatrix} R_{11}C_1 & R_{12}C_2 & \cdots & R_{1M}C_M \\ R_{21}C_1 & R_{22}C_2 & \cdots & R_{2M}C_M \\ \vdots & \vdots & \vdots & \vdots \\ R_{M1}C_1 & R_{M2}C_2 & \cdots & R_{MM}C_M \end{bmatrix} \quad (37)$$

It becomes clear now why matrix \mathbf{T} is known as the time-constant matrix (or RC-matrix). Its elements are RC products that can be computed directly from the circuit.

Valuable information can be obtained from this matrix. For instance, classical hand analysis techniques formulate the poles of circuits as the inverse of the capacitance at each circuit node multiplied by the impedance seen at that node. Matrix \mathbf{T} gives a more precise definition and the conditions under which it is valid. A pole can be considered to be associated to one node if the (trans)resistances of the other ports to that port are negligible.

4.3. Approximate root equations

The sum of products of the eigenvalues of matrix \mathbf{T} are related to its k -th trace by:

$$\sum_{i_1 < i_2 < \cdots < i_k} \tau_{i_1} \cdot \tau_{i_2} \cdot \cdots \cdot \tau_{i_k} = (-1)^k T_k \quad (38)$$

where the k -th trace of matrix \mathbf{T} is defined as:

$$T_k = \sum_{\alpha_1 < \dots < \alpha_k} \begin{bmatrix} R_{\alpha_1 \alpha_1} & R_{\alpha_1 \alpha_2} & \dots & R_{\alpha_1 \alpha_k} \\ R_{\alpha_2 \alpha_1} & R_{\alpha_2 \alpha_2} & \dots & R_{\alpha_2 \alpha_k} \\ \vdots & \vdots & \vdots & \vdots \\ R_{\alpha_k \alpha_1} & R_{\alpha_k \alpha_2} & \dots & R_{\alpha_k \alpha_k} \end{bmatrix} C_{\alpha_1} \dots C_{\alpha_k} \quad (39)$$

Let us consider the equations relating eigenvalues to the first two traces:

$$\begin{aligned} \sum_{i=1}^M \tau_i &= -T_1 \\ \sum_{i=1}^{M-1} \sum_{j=i+1}^M \tau_i \cdot \tau_j &= T_2 \end{aligned} \quad (40)$$

If the first two poles are sufficiently separated, *i.e.*, the following two conditions are satisfied:

$$\begin{aligned} \tau_1 + \tau_2 &\gg \sum_{i>2} \tau_i \\ \tau_1 \cdot \tau_2 &\gg \sum_{i>2} \sum_{j>i} \tau_i \cdot \tau_j \end{aligned} \quad (41)$$

the following set of equations can be approximated

$$\begin{aligned} \tau_1 + \tau_2 &\approx -T_1 \\ \tau_1 \cdot \tau_2 &\approx T_2 \end{aligned} \quad (42)$$

If these two poles are close or constitute a complex conjugate pair, the following expression is easily obtained for the poles:

$$p_{1,2} \approx -\frac{1}{2} \cdot \frac{T_1}{T_2} \cdot (1 \pm \sqrt{1 - 4 \cdot \frac{T_2}{T_1^2}}) \quad (43)$$

However, if the two poles are splitted enough *i.e.* $\tau_1 \gg \tau_2$, a further approximation can be applied and the following expressions for p_1 and p_2 are obtained:

$$\begin{aligned} p_1 &\approx -\frac{1}{T_1} \\ p_2 &\approx -\frac{T_1}{T_2} \end{aligned} \quad (44)$$

The same approach can be applied recursively to higher frequency poles. Let us assume one port with a large time constant. This means that this pole remains practically uncharged for high-frequency signals, that is, it behaves like a short-circuit. Therefore, the circuit resulting after the contraction of the capacitor associated to the port with the highest time-constant will have approximately the same higher frequency poles that the original one. This transformation yields a shift towards zero

of the dominant pole whereas the remaining ones are kept in approximately the same locations. Therefore, once the contraction has been performed, the same extraction procedure can be applied to higher frequency poles.

4.4. Symbolic calculation of RC factors

Equations (43) and (44) approximate the pole positions as a function of the first two traces, given as:

$$T_1 = \sum_{\alpha=1}^M R_{\alpha\alpha} \cdot C_{\alpha}$$

$$T_2 = \sum_{\alpha=1}^{M-1} \sum_{\beta=\alpha+1}^M \begin{bmatrix} R_{\alpha\alpha} & R_{\alpha\beta} \\ R_{\beta\alpha} & R_{\beta\beta} \end{bmatrix} \cdot C_{\alpha} \cdot C_{\beta}$$
(45)

Then, approximate expressions for these two traces are obtained in several steps. First, a simplification before generation step is applied for the circuit capacitors, by eliminating the capacitors that have an influence in T_1 and T_2 such that their impact on each pole position is below an error threshold. The effect of this simplification before generation step is that the number of addends in (45) is significantly reduced and only a reduced number of (trans)resistances has to be calculated.

In a second step, a circuit is considered for the calculation of each (trans)resistance in (45). Each of these circuits is a purely resistive circuit obtained from that resulting from the previous step by removing all capacitors and applying the appropriate excitation source for each (trans)resistance. For illustration's sake, **Fig. 5** shows the transformations on the original circuit for the calculation of the transresistances of a circuit with two capacitors: C_{α} and C_{β} . These resistive circuits can still be relatively complex and therefore also suffer from the exponential growth of symbolic results with the circuit size. Therefore, first a simplification before generation technique is applied. For reasons that will become clear below, the SBG technique works at the circuit level, by contracting nodes and eliminating devices whose contribution to the pole position is negligible. A separate circuit is considered for each (trans)resistance, so that the simplest circuit for each of them can be obtained.

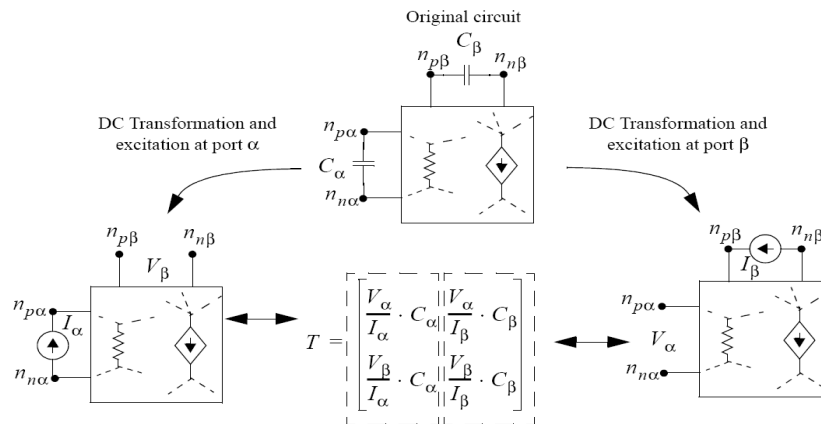


Figure 5: Illustrating circuit transformation for calculation of (trans)resistances.

Then, a simplification during generation technique is applied to get the dominant terms of each (trans)resistance. The simplification during generation technique applied is based on the two-graph approach and is able to generate symbolic terms in decreasing order of magnitude one-by-one. Initially, the dominant term of each (trans)resistance is generated and queued. The dominant term of each trace is considered and that with the largest influence on the pole or poles considered is selected for the symbolic pole expression. The following term for the (trans)resistance that supplied the selected symbolic term is then generated.

If the errors specs are not met, the next term in the list has to be added to T_1 or to T_2 . At each iteration two ratios are used to decide if the next term in T_1 or that in T_2 should be collected. Assuming that F terms have already been added into T_1 and G terms into T_2 , the two ratios are:

$$r_1 = \frac{\sum_{k=1}^F R_{\alpha\alpha}^{(k)} \cdot C_{\alpha}^{(k)}}{\sum_{i=1}^M R_{ii} \cdot C_i} \quad (46)$$

$$r_2 = \frac{\sum_{k=1}^G \begin{bmatrix} R_{\alpha\alpha}^{(k)} & R_{\alpha\beta}^{(k)} \\ R_{\beta\alpha}^{(k)} & R_{\beta\beta}^{(k)} \end{bmatrix} \cdot C_{\alpha}^{(k)} \cdot C_{\beta}^{(k)}}{\sum_{i=1}^{M-1} \sum_{j=i+1}^M \begin{bmatrix} R_{ii} & R_{ij} \\ R_{ji} & R_{jj} \end{bmatrix} \cdot C_i \cdot C_k}$$

These equations represent the ratios between the terms added to T_1 and T_2 up to that iteration and the exact values given by the equations in (45). The next term in the trace with the smallest ratio is added and the errors specs checked again. The generation procedure continues until the error specs are met.

4.5. Extraction of zeros

The modification-decomposition method can also be applied to the zero extraction problem by performing some modifications on the time-constants matrix, leading to a modified time-constants matrix, from which the numerical zero spectrum can be computed by means of the QR algorithm. However, unlike the time-constants matrix, although the elements of the modified time-constants matrix are dimensionally RC products, they cannot be easily interpreted in terms of circuit (trans)resistances and capacitors. Therefore, the approximated formulas cannot be directly applied.

A solution arises from the application of feedback systems theory. Let us consider the system in **Fig. 6**, where $H(s)$ corresponds to the transfer function of the system whose zeros are to be obtained. The transfer function of the complete feedback system is:

$$H_{fb}(s) = \frac{k \cdot D(s)}{D(s) + k \cdot N(s)} \quad (47)$$

If $k \rightarrow \infty$, then (47) becomes:

$$H_{fb}(s) = \frac{D(s)}{N(s)} \quad (48)$$

Therefore, the poles of the feedback system correspond to the zeros of the original one.

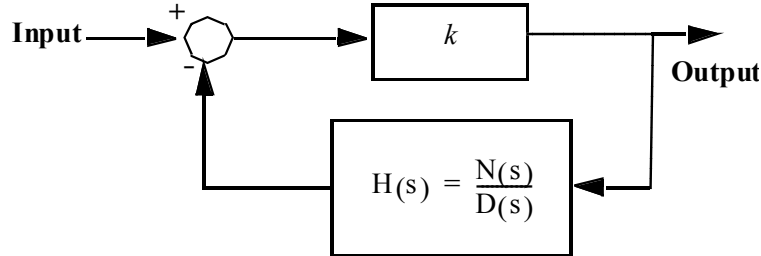


Figure 6: Block diagram.

Dealing with the poles of this feedback system is not a complex task. The forward gain k does not depend on the complex frequency variable. Therefore, it contributes to the different (trans)resistances of the matrix T . As stated above, simplification before and during generation techniques are applied to the different (trans)resistances. Simplification before generation techniques are applied normally but the element corresponding to the forward gain k is preserved as the application of $k \rightarrow \infty$ prevents its elimination. The simplification during generation technique used is based on the enumeration of common spanning trees in the two-graph approach in decreasing order of magnitude. Applying $k \rightarrow \infty$ reduces to imposing that the element corresponding to k must belong to the spanning tree⁴.

4.6. Experimental results

As an application example let us consider the uA741 operational amplifier in **Fig. 4**. The approach above provides the following expressions for the first two poles and the first zero:

$$\begin{aligned}
 p_1 &= -\frac{g_{m1}g_{m3} + g_{m1}g_{m4} + g_{m3}g_{m4}}{C_c g_{m1}g_{m4}g_{m16}g_{m17}r_{o13B}r_{o17}r_{\pi16}r_{\pi17}} \\
 p_2 &= -\frac{g_{m1}g_{m3} + g_{m1}g_{m4}}{R_{11}(C_{\pi22} + C_{\pi24})(g_{m1}g_{m3} + g_{m1}g_{m4} + g_{m3}g_{m4})} \\
 z_1 &= -\frac{g_{m10}(g_{m5} + g_{m6})R_4}{R_{11}[(1 + g_{m10}R_4)(g_{m5} + g_{m6} + g_{m5}g_{m6}(R_1 + R_2))(C_{\pi22} + C_{\pi24})]}
 \end{aligned} \quad (49)$$

If we now consider the BiCMOS operational amplifier in **Fig. 7**, the approach provides symbolic expressions for the four poles:

⁴ Refer to Chapter 7 for detailed descriptions of simplification before and during generation techniques.

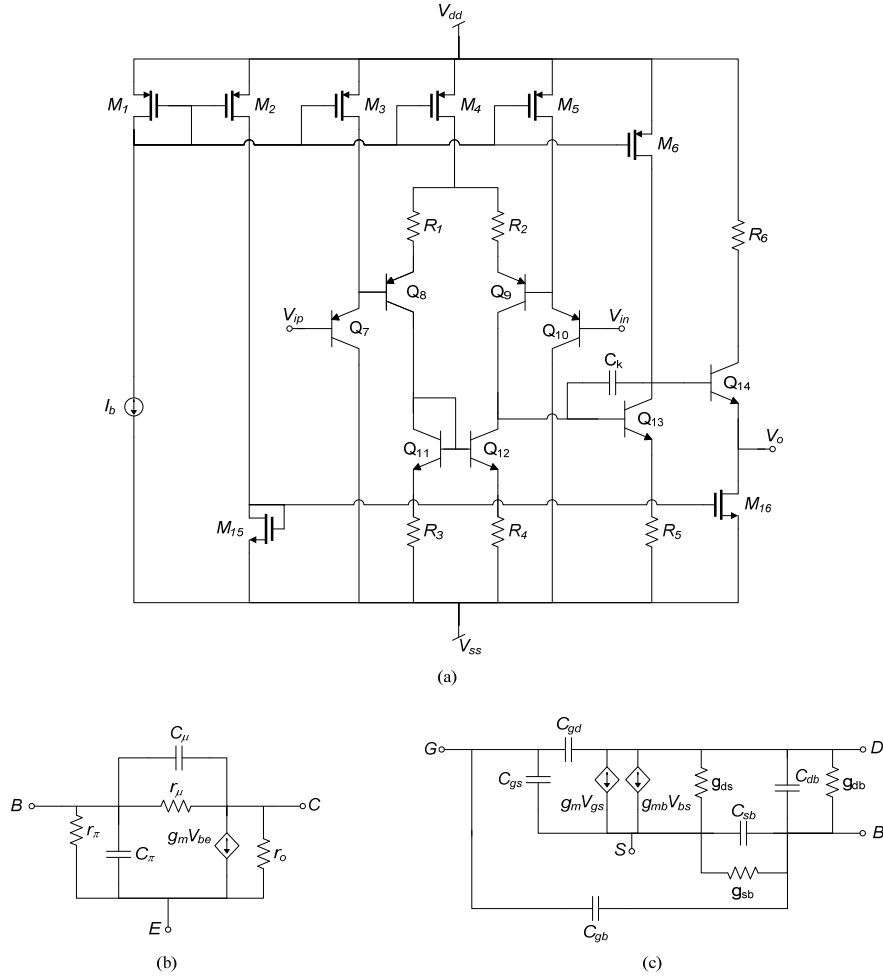


Figure 7: (a) BiCMOS operational amplifier; (b) bipolar transistor model; and (c) MOS transistor model.

$$\begin{aligned}
 p_1 &= -\frac{g_{ds6}(R_1 + R_2)[g_{m12}R_4(g_{m1} + g_{ds1}) + g_{m1}]}{r_{\pi13}g_{m1}g_{m12}g_{m13}R_1R_4(C_k + C_{\mu13})} \\
 p_2 &= -\frac{g_{m7}g_{m10}}{g_{m7}C_{\pi10} + g_{m10}C_{\pi7}} \\
 p_3 &= -\frac{g_{m8}g_{m10}[g_{m8}(R_1 + R_2)(g_{m12}R_4(g_{m1} + g_{ds1}) + g_{m1}) + g_{m1}g_{m12}R_4]}{[g_{m8}g_{m9}(R_1 + R_2)(g_{m12}R_4(g_{m1} + g_{ds1}) + g_{m1}) + g_{m1}g_{m12}R_4(g_{m8} + g_{m9})]C_{\pi10}} \\
 p_4 &= -\frac{g_{m10}[g_{m12}(R_1 + R_2)R_4(g_{m1} + g_{ds1}) + g_{m1}R_2]}{g_{m8}g_{m9}g_{m12}R_4(g_{m1}(R_1 + R_2) + g_{ds1}R_1)C_{\pi10}}
 \end{aligned} \tag{50}$$

5. Conclusion

Poles and zeros include essential information for circuit design. Therefore, the extraction of symbolic expressions for poles and zeros has attracted the attention of symbolic analysis researchers. First reported techniques imitated the usual techniques applied for hand analysis: elimination of unimportant terms in symbolic network

functions (considered simplification after generation techniques in the classification used in this chapter), application of root splitting techniques and extraction of roots of low-order symbolic polynomials. These techniques also inherited the exponential growth of symbolic results with the circuit size. Therefore, more recent analysis techniques have incorporated simplification before and during generation techniques. An essential component of the symbolic pole/zero extraction technique is the error control mechanism. Some techniques have used the same approximation techniques used for symbolic network equations, based on the control of magnitude and phase errors. However, a well-controlled magnitude and phase error does not necessarily imply accurate symbolic pole-zero solutions. Therefore, more recent techniques have focused on the development of analysis techniques together with error control mechanisms specifically devoted to guarantee the desired accuracy on symbolic pole/zero expressions.

Acknowledgements

The preparation of this chapter has been partially supported by the TIC-2532 Project, funded by Consejería de Innovación, Ciencia y Empresa, Junta de Andalucía and by the Project TEC2010-14825/MIC funded by the Spanish Ministry of Science and Innovation with support from ERDF. Dr. C. Sánchez-López thanks the support of the JAE-Doc program of CSIC, co-funded by FSE.

References

- [1] F.V. Fernández, A. Rodríguez-Vázquez, and J.L. Huertas, "A tool for symbolic analysis of analog integrated circuits including pole/zero extraction", in *10th European Conference on Circuit Theory and Design*, 1991, pp. 752-761.
- [2] F.V. Fernández, A. Rodríguez-Vázquez and J.L. Huertas, "Interactive AC modelling and characterization of analog circuits via symbolic analysis", *Analog Integrated Circuits and Signal Processing*, vol. 1, no. 3, pp. 183-208, Nov. 1991.
- [3] G. Gielen, H. Walscharts and W. Sansen, "ISAAC: a symbolic simulator for analog integrated circuits", *IEEE Journal of Solid-State Circuits*, vol. 24, pp. 1587-1597, Dec. 1989.
- [4] G. Nebel, U. Kleine and H. Pfeleiderer, "Symbolic pole/zero calculation using SANTAFE", *IEEE Journal of Solid-State Circuits*, vol. 30, no. 7, pp. 752-761, July 1995.
- [5] J. Vlach and K. Singhal, *Computer methods for circuit analysis and design*, 2nd ed. Van Nostrand Reinhold, 1993.
- [6] G. H. Golub and C. F. Van Loan, *Matrix computations*, 3rd ed. Johns Hopkins, 1996.
- [7] J.-J. Hsu and C. Sechen, "Accurate extraction of simplified symbolic pole/zero expressions for large analog IC's", in *IEEE International Symposium on Circuits and Systems*, 1995, pp. 2083-2087.
- [8] G. Droge, T. Czysz and E.-H. Horneber, "Symbolic pole and zero estimation for circuit design", in *IEEE International Conference on Electronics, Circuits and Systems*, 1996, pp. 93-97.
- [9] E. Hennig, "Matrix approximation techniques for symbolic extraction of poles and zeros", *Analog Integrated Circuits and Signal Processing*, vol. 31, pp. 81-100, 2002.
- [10] J.-J. Hsu and C. Sechen, "Fully symbolic analysis of large analog integrated circuits", in *IEEE Custom Integrated Circuits Conference*, 1994, pp. 457-460.
- [11] O. Guerra, J.D. Rodríguez-García, F.V. Fernández and A. Rodríguez-Vázquez, "A symbolic pole/zero extraction methodology based on analysis of circuit time-constants", *Analog Integrated Circuits and Signal Processing*, vol. 31, pp. 101-118, 2002.
- [12] S. B. Haley, "The generalized eigenproblem: pole-zero computation", *Proceedings of the IEEE*, vol. 76, no. 2, pp. 103-120, Feb. 1988.
- [13] S. B. Haley, "Modification-decomposition transformation in analog design", *International Journal of Computer Aided VLSI Design*, pp. 407-428, 1991.

CHAPTER 12**Automatic Nonlinear Behavioral Model Generation Using Symbolic Circuit Analysis****Ralf Sommer^{*}, Eckhard Hennig, Gregor Nitsche, Jochen Broz and Peter Schwarz***Ilmenau University of Technology and Institute for Microelectronic and Mechatronic Systems GmbH, Germany*

Abstract: The aim of symbolic analysis that has its origin in the design of analog circuits is the extraction of dominant system behavior by automated derivation of approximated symbolic formulas. Since exact symbolic analysis will yield exceptionally complex expressions even for rather small systems a class of symbolic approximation techniques has been developed that allow a reduction of the complexity of symbolic equations and their later solution by means of mixed symbolic and numerical strategies. Hence, it becomes possible to reduce the underlying nonlinear Differential-Algebraic systems of Equations (DAE systems) of component-based networks and systems to a behavioral description of a predefined accuracy. So it is a major advantage of the approach that the model simplification is performed by an automatic error control and that the simplified models are physically interpretable again. The contribution will give an overview of the symbolic tool *Analog Insydes* algorithms for extraction of dominant behavior of linear systems, as well as algorithms for generating behavioral models from nonlinear DAEs. Moreover, the underlying methodology has been extended to the application of analysis and modeling in non-electrical domains (e.g. gas-pipeline nets) and for multi-physical (e.g. mixed electrical and mechanical) systems. For the latter a library was developed in cooperation with the *Fraunhofer IIS/EAS* for symbolic models of micro-mechanical elements that can be connected to networks including electrical components as well.

Keywords: behavioral modeling, automated behavioral model generation, symbolic analysis, nonlinear symbolic circuit equation, DAE system (differential algebraic equations), MNA (modified nodal analysis), STA (sparse tableau analysis), lumped element representation, symbolic approximation, model reduction, term cancellation, term ranking, multi-physical system, heterogeneous system, conservative system, electrical system, mechatronic system, micromechanical acceleration sensor, Analog Insydes.

1. Introduction

With the decreasing structural size going along with expanding complexity of technical systems there is an emerging demand for new design methods and modeling support. This becomes even more important because of the increasing heterogeneity of technical systems. In particular, for the design of mechatronic systems this leads to the following problem: There are established tools for the design of the mechanical or electrical parts like FEM (Finite Element Method) [1], multi-body or circuit simulators, but those are usually specialized on their physical domain. Therefore, consideration of interactions between mechanical and electrical components is extraordinarily laborious. These interactions often have to be taken into account because the assembly of independently optimized subcomponents usually does not lead to an optimal system. On the other side, considering coupling effects results in new challenges in many aspects, which

^{*}Address correspondence to Ralf Sommer: Ilmenau University of Technology and Institute for Microelectronic and Mechatronic Systems GmbH, Germany; E-mail: ralf.sommer@imms.de

lead from the need of designers competence in multiple disciplines (electrical and electronic as well as mechanical engineering, physics and mathematics) up to the high complexity of mathematical models which demand the employment of adapted simulation tools.

Such software has to be capable of dealing with the multi-physical aspects of such systems. There are several suitable modeling languages like VHDL-AMS [2] or Modelica [3] and corresponding simulators like Virtuoso® AMS Designer, AdvanceMS™ or Dymola™ available. They allow for a modularized modeling of the complete system or parts of it with arbitrary accuracy. But the modeling process of heterogeneous systems is very time consuming and, moreover, the resulting mathematical models become very complex even for comparatively small systems, raising numerical problems with respect to robustness, efficiency, and stability.

Today the simulation of industrial-sized systems lies beyond the limits of this approach. In order to reduce the numerical effort, model reduction techniques become more and more important. In this article, we present a modeling approach, which is based on symbolic methods and can be adapted to multi-physical systems due to its general mathematical principle. This includes an automatic generation of behavioral models including model reduction for electrical as well as for mechatronic components and a combination of both. Such generated models allow an interactive processing and a more efficient simulation of the overall system, thanks to their reduced complexity. This may even enable application of optimization and control methods.

2. Bottom-Up Behavioral Modeling Using *Analog Insydes*

Automatic bottom-up model generation by symbolic analysis of circuit netlists and/or equations allows deriving accurate behavioral models. The model generation process in this contribution is based on the symbolic analysis tool *Analog Insydes* [4, 5]. This powerful tool offers functionality to automatically set up circuit equations for a circuit from its netlist and to use them as foundation for behavioral model generation. Furthermore, an additional input for DAE (differential algebraic equation) systems could be implemented to handle systems modeled by behavioral descriptions instead of circuit netlists and to support modeling of multi-physical systems. **Fig. 1** visualizes an exemplary process as it would be performed in bottom-up model generation.

Whereas the number of the circuit equations is increasing proportional to the circuit's number of nodes, their complexity is again amplified by the complexity of the applied device models. Therefore, the achieved equation sets are often extremely complex - impossible to be set up by hand. Model reduction methods can be applied to reduce the complexity of the equations by term reduction techniques [6-9] as it will be described in section 4.3. Another benefit of this symbolic approximation technique is to ensure a user-specified accuracy - hence, this is one of the very few modeling methods that allow satisfying a predefined accuracy.

As the equations' complexity decreases but the resulting error grows with the degree of model reduction, it is up to the user to find a suitable trade-off between size, interpretability and accuracy of the model. Experiments show that for reasonable error margins of 5 to 10% the complexity can be reduced by a factor of 10 to 100, resulting in a good speed-up of simulation.

Finally, the behavioral model can be generated from DAEs using *Analog Insydes'* model export function, which supports model creation for almost every behavioral

simulator by the inclusion of several AHDLs (analog hardware description languages) - e.g. VHDL-AMS, Verilog-A, MAST.

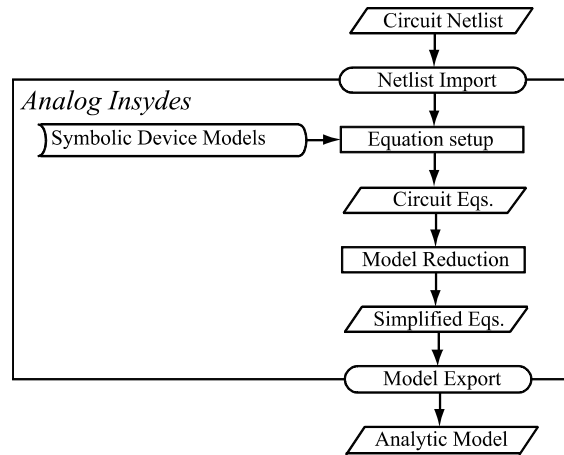


Figure 1: Bottom-up modeling process.

3. Symbolic Equation Formulation of Nonlinear Dynamic Circuits and Systems

In this section an outline of circuit theory [10, 11] will be given in order to introduce into the automatic derivation of linear and nonlinear circuit and system equations, which is the basis for automated generation of behavioral models and model order reduction. Electrical but also multi-physical systems have in common that they consist of a finite number of components or elements that are linked together by connecting multiports. Each component can be modeled by the mathematical relations of its corresponding physical branch quantities. These quantities can be divided into 2 classes, namely *through* and *across* quantities. *Effort* or *across* quantities are quantities that are measured between two points within the system or with reference to a predefined reference value. *Flow* or *through* quantities represent forces or charge/material transport through a component or along connections between them, which may be made visible only by opening the link. The product of the corresponding *through* and *across* quantities has the dimension of power (Table 1).

Table 1: Corresponding *through* and *across* quantities in different energy domains.

Energy Domain	<i>Through</i> Quantity φ	<i>Across</i> Quantity δ	Power P [W]
Electrical	<i>Current</i> i [A]	<i>Voltage</i> u [V]	$P = ui$
Mechanical (translational)	<i>Force</i> F [N]	<i>Velocity</i> v [m/s]	$P = Fv$
Mechanical (rotational)	<i>Torque</i> τ [Nm]	<i>Angular Velocity</i> ω [rad/s]	$P = \tau\omega$
Hydraulic	<i>Flow Rate</i> J [m ³ /s]	<i>Pressure</i> p [N/m ²]	$P = pJ$
Thermal	<i>Entropy Flux</i> S [J/Ks]	<i>Temperature Difference</i> ΔT [K]	$P = S\Delta T$

For practical reasons in mechanical and thermal systems often alternative *through* and *across* quantities are used, which products do not directly yield power as shown in **Table 2**.

Exploiting the analogies to electronic circuits any kind of *through* and *across* quantities can be handled similar to currents and voltages in the context of network analysis. As systems have to be considered in three spatial dimensions \mathbf{x} , \mathbf{y} and \mathbf{z} for some physical domains (e.g. mechanics) such *through* and *across* quantities have to be vector-type (**Table 3**) in contrast to the one-dimensional descriptions of **Table 1**.

Table 2: Alternative *through* and *across* quantities for mechanical systems.

Energy Domain	<i>Through</i> Quantity Φ	<i>Across</i> Quantity δ	Power P [W]
Mechanical (translational)	<i>Force</i> F [N]	<i>Displacement</i> λ [m]	$P = F \, d\lambda / dt$
Mechanical (rotational)	<i>Torque</i> τ [Nm]	<i>Angle</i> α [rad]	$P = \tau \, d\alpha / dt$

A key for setting up equations of any system having *through* and *across* quantities are the *cutset law* for *through* and the *mesh law* for *across* quantities, in electrical domain *Kirchhoff's Current Law* (KCL) and *Kirchhoff's Voltage Law* (KVL). These laws are characteristic for conservative systems meaning that there is no dissipation of energy so that the total energy remains constant. These characteristics are independent of the circuits' element relations and can be rooted to some interesting properties of the underlying graphs and their incidence matrices (e.g. proof of *Tellegen's theorem*). This will be motivated and briefly described in the next sections.

To process spatially distributed mechanical systems with the following network-oriented modeling approach of electrical systems, a discretized system consideration is necessary. This can be obtained from a manual or automatical discretization method like FEM [1], FDM (Finite Differences Method) or BEM (Boundary Elements Method) or from multibody modeling with only discrete, lumped elements (e.g. beams, plates and membranes) (**Fig. 2**).

Table 3: Vector-type *through* and *across* variables.

Domain	<i>Through</i> Variables	<i>Across</i> Variables
Electronics	<i>Current</i> i	<i>Voltage</i> i
Mechanics	<i>Force</i> $\mathbf{F} = (F_x, F_y, F_z)$	<i>Displacement</i> $\boldsymbol{\lambda} = (\lambda_x, \lambda_y, \lambda_z)$
	<i>Torque</i> $\boldsymbol{\tau} = (\tau_x, \tau_y, \tau_z)$	<i>Rotation</i> $\boldsymbol{\alpha} = (\alpha_x, \alpha_y, \alpha_z)$
Hydraulics	<i>Mass Flow</i> J	<i>Pressure</i> p

Because of the generic branch-oriented description of network elements and the topological description of the connecting network *via* incidence matrices, the connections of discrete, lumped components correspond for different domains, so that the following approach can be applied to different domains (e.g. electrical, mechanical) and multi-domain modeling (e.g. mechatronics).

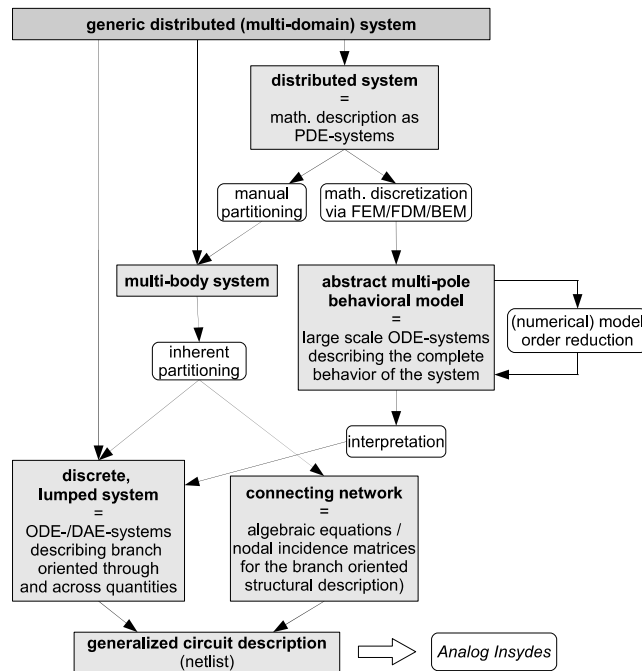


Figure 2: Handling of generic distributed (multi-domain) systems as generalized Kirchhoff networks.

3.1. Element relations

The precondition of the following annotations is the consideration of lumped networks or systems only. Here, element relations of arbitrary algebraic or Differential Mathematical Relations (DAE) without restriction to only one dimension are allowed, *i.e.* generic n -port elements. As a simple example of an electrical one-port relation a diode – with its element relation written in voltages and currents as well as in *through* and *across* variables – is shown in **Fig. 3**.

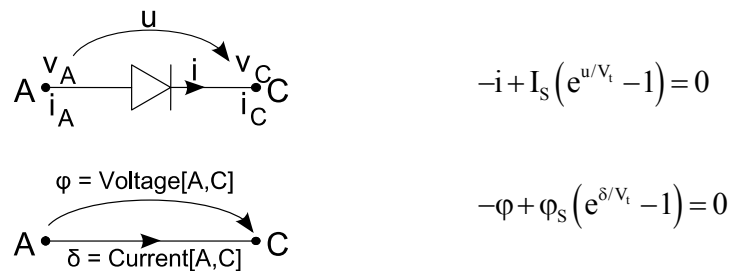


Figure 3: Single-/one-port (-branch) element: Diode.

For the quantity representation a branch oriented description is used as it is illustrated in **Figs. 3** and 4, facilitating the use of vector-type *through* and *across* quantities. Elements with dimensional dependent (multi-dimensional) quantities can be represented as n -port or n -pole that are mathematically equivalent. In this contribution we will refer to the n -port representation (left side in **Fig. 4**).

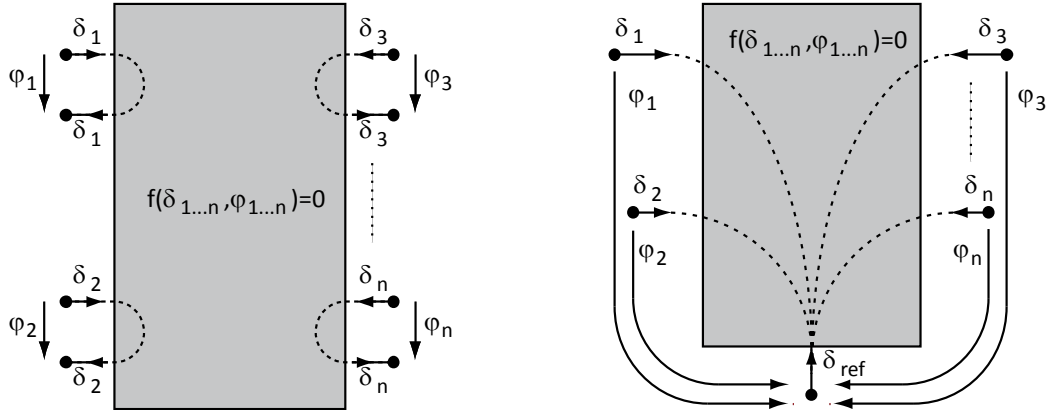


Figure 4: General n-port element with general relation between *through* (φ) and *across* (δ) quantities (left) and corresponding n-pole representation (right).

Likewise, non-electrical elements of other domains can be modeled in a similar way. Considering a mechanical beam element (**Fig. 5**), the connection points (pins) “1” and “2” to the neighboring elements have to be defined. At each pin, the mechanical quantities have to be considered in all spatial directions x , y and z . So the beam element’s *through* quantities are 3-dimensional forces \mathbf{F} and torques $\boldsymbol{\tau}$ instead of one-dimensional currents, and its *across* quantities are 3-dimensional displacements $\boldsymbol{\nu}$ and rotations $\boldsymbol{\omega}$ instead of one-dimensional voltages (**Fig. 5**).

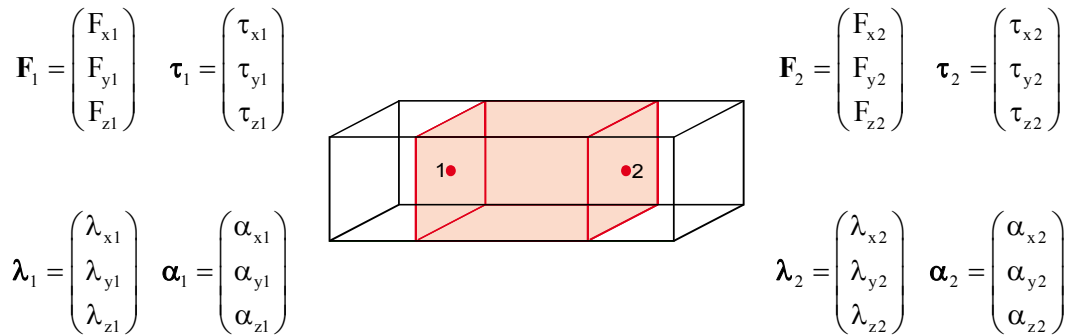


Figure 5: Beam quantities for a lumped, mechanical beam element.

Thus, the beam element can be considered as a multi-branch or multi-port element using scalar quantities for each spatial direction as well as an equivalent single-branch element using vector-type quantities (**Fig. 6**).

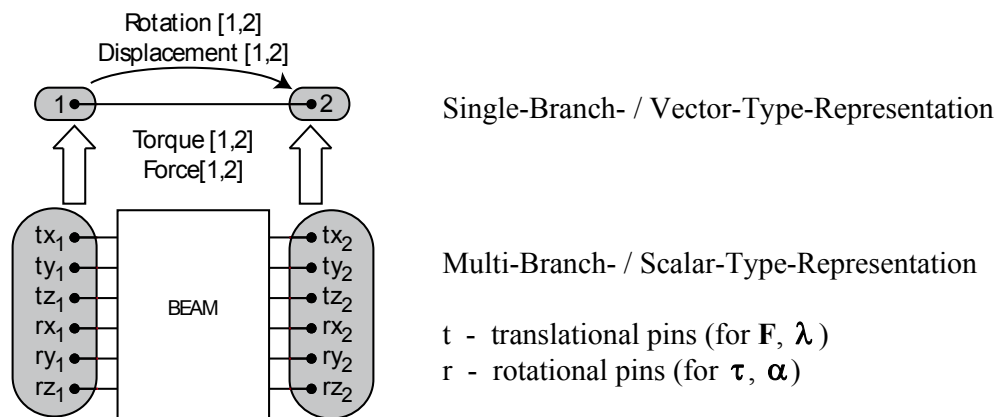


Figure 6: Branch definition for a lumped, mechanical beam element.

t - translational pins (for \mathbf{F} , $\boldsymbol{\lambda}$)
r - rotational pins (for $\boldsymbol{\tau}$, $\boldsymbol{\alpha}$)

In general, the beam is mathematically described by Partial Differential Equations (PDEs). Only for small extension in length direction, the beam may be approximately considered as a lumped element, as typical in Finite Element Analysis. Its dynamic behavior, *i.e.* the relations of *through* and *across* quantities, is then specified in the form of Ordinary Differential Equations (ODEs) in vector or scalar form. In the case of a linear beam element - which behaves linearly as a spring (*Hooke's Law*) concerning small flexures and torsions only - the equation's structure is:

$$\boldsymbol{\varphi}^T = -(\mathbf{M}\ddot{\boldsymbol{\delta}}^T + \mathbf{D}\dot{\boldsymbol{\delta}}^T + \mathbf{K}\boldsymbol{\delta}^T) \quad (1)$$

where \mathbf{M} is the mass matrix, \mathbf{D} the damping matrix, and \mathbf{K} the stiffness matrix of the finite elements. Furthermore, the vectors $\boldsymbol{\varphi}$ of the *through* variables (forces \mathbf{F} and torques $\boldsymbol{\tau}$) and the corresponding vector $\boldsymbol{\delta}$ of the *across* quantities (displacements $\boldsymbol{\lambda}$ and torsions $\boldsymbol{\alpha}$) in the pins "1" and "2" are:

$$\boldsymbol{\varphi}^T = (\mathbf{F} \quad \boldsymbol{\tau})^T = (F_{x1}, F_{y1}, F_{z1}, F_{x2}, F_{y2}, F_{z2}, M_{x1}, M_{y1}, M_{z1}, M_{x2}, M_{y2}, M_{z2})^T \quad (2)$$

$$\boldsymbol{\delta}^T = (\boldsymbol{\lambda} \quad \boldsymbol{\alpha})^T = (\lambda_{x1}, \lambda_{y1}, \lambda_{z1}, \lambda_{x2}, \lambda_{y2}, \lambda_{z2}, \alpha_{x1}, \alpha_{y1}, \alpha_{z1}, \alpha_{x2}, \alpha_{y2}, \alpha_{z2})^T \quad (3)$$

Due to the vector property the port behavior of an element with two pins (*e.g.* a beam element) is, therefore, described by a system of 12 equations. These equations describe the lumped beam element behavior in a local reference system strictly connected to the individual beam element. In a "network" with many beam elements having different spatial coordinates and directions, the quantities (node variables) of the global reference system have to be transformed into the different individual local quantities using the orthogonal transformation matrices \mathbf{T} and \mathbf{T}^{-1} yielding:

$$\boldsymbol{\varphi}^T = -(\mathbf{T}^{-1}\mathbf{M}\mathbf{T}\ddot{\boldsymbol{\delta}}^T + \mathbf{T}^{-1}\mathbf{D}\mathbf{T}\dot{\boldsymbol{\delta}}^T + \mathbf{T}^{-1}\mathbf{K}\mathbf{T}\boldsymbol{\delta}^T) \quad (4)$$

Thereby the mechanical law of the continuity of the displacements in the nodes corresponds to *Kirchhoff's voltage law* and the principle of *d'Alembert* corresponds to *Kirchhoff's current equations*.

To handle spatially distributed systems in a similar way, they have to be partitioned into a discrete number of lumped basic elements, which can be described by ODEs (**Fig. 5**). According to numerical experiments, often the separation into 4 or 6 elements already leads to satisfying accurate multi-body models [12]. **Fig. 7** illustrates this procedure by the example of a "long" mechanical beam (to be described by PDEs) which is split into a chain of "small" basic beam elements, corresponding to a network of basic beam models (each of them is described by ODEs).

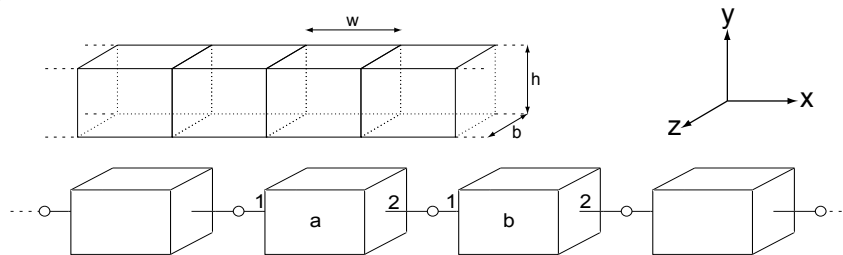


Figure 7: Partitioning of a mechanical beam into a multi-body network model with basic beam elements.

Kirchhoff's voltage and current laws are valid also for the interconnection points of two adjacent beam elements. In the case of two connected beam segments *a* and *b* whose "left" and "right" sides are denoted by the indices "1" and "2" (see also Fig. 5) the following equations hold:

$$\begin{aligned}
 \lambda_{x1}^a &= \lambda_{x2}^b, & \lambda_{y1}^a &= \lambda_{y2}^b, & \lambda_{z1}^a &= \lambda_{z2}^b \\
 F_{x1}^a + F_{x2}^b &= 0, & F_{y1}^a + F_{y2}^b &= 0, & F_{z1}^a + F_{z2}^b &= 0 \\
 \alpha_{x1}^a &= \alpha_{x2}^b, & \alpha_{y1}^a &= \alpha_{y2}^b, & \alpha_{z1}^a &= \alpha_{z2}^b \\
 \tau_{x1}^a + \tau_{x2}^b &= 0, & \tau_{y1}^a + \tau_{y2}^b &= 0, & \tau_{z1}^a + \tau_{z2}^b &= 0
 \end{aligned} \tag{5}$$

Although the resulting DAE system is comparatively large it is well suitable for the application of symbolic analysis because the quite small number of coefficients which equal not zero depend only on a few material properties – e.g. the density ρ , the elastic modulus E and the Young's modulus G – and geometric dimensions, as the width w , the breadth b and the height h (Fig. 8).

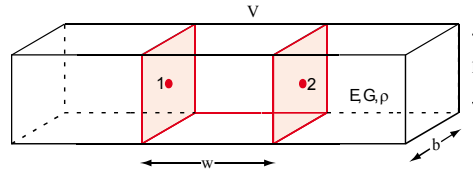


Figure 8: Geometrical dimensions and material properties of a beam element.

As another example Fig. 9 sketches a motor as a two-port element on the left, with current and torque as *through* quantities and voltage and angular velocity as *across* quantities. The corresponding n-port and the resulting linear DAE system of the element relations are shown on the right side. Note that for a detailed consideration the mechanical *through* and *across* quantities τ_2 and ω_2 are vector-type quantities again.

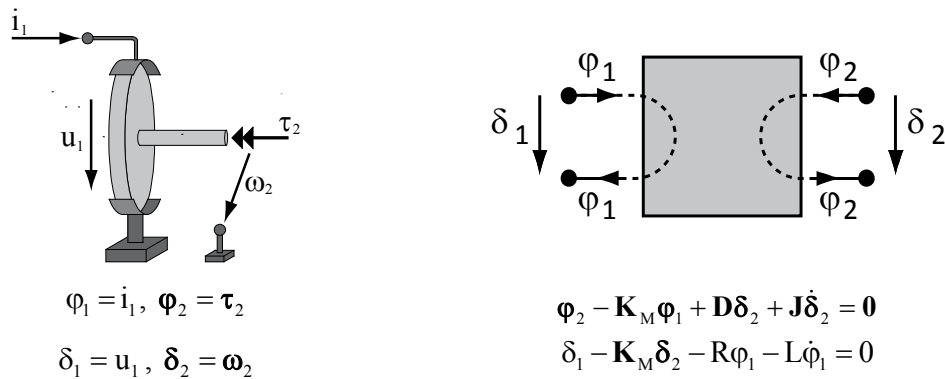


Figure 9: A motor as two-port element and its element relations.

The idealized relations between electrical and mechanical quantities of the motor are:

$$u_1 = \mathbf{K}_M \omega_2, \quad \tau_2 = \mathbf{K}_M i_1 \tag{6}$$

Taking into account the electrical resistance R and the inductance L as well as the mechanical-rotational parameters damping D and inertia J , the element relations are:

$$u_1 = R i_1 + L \frac{di_1}{dt} + \mathbf{K}_M \omega_2 \quad \text{and} \quad \mathbf{M}_2 + \mathbf{D} \omega_2 + \mathbf{J} \frac{d\omega_2}{dt} = \mathbf{K}_M i_1 \tag{7}$$

Although the introduced considerations of system elements mainly focus on the systems' operation regions regarding only the linear behavior and components' parameters ("small signal behavior"), the extension to nonlinear behavior and geometrical nonlinearities is possible.

3.2. Connecting graph and network topology

In addition to the elements the topology of the circuit has to be described representing all connections between the system's components as a graph. Because of the generality of the basic principles of *cutset law* and *mesh law* for all domains of conservative systems and because of the consistently branch-oriented description of all *through* and *across* relations the following generalized network analysis methodology can be applied. Hence, the system of analysis equations can be set up in the same manner for electrical, mechanical, hydraulical or thermal systems.

In **Fig. 10** the connecting network (red) links the n ports of the components to form a conservative system. This connecting network can easily be abstracted to a mathematical graph structure, consisting only of the connected nodes v_k and the elements' branches b_k ($k \in [1, \dots, n]$), weighted by *through* (φ_k) and *across* (δ_k) quantities (**Fig. 11**).

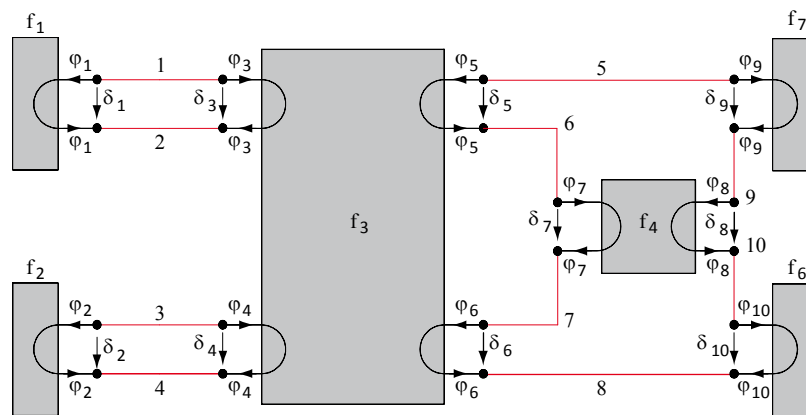


Figure 10: Connecting network between 6 components.

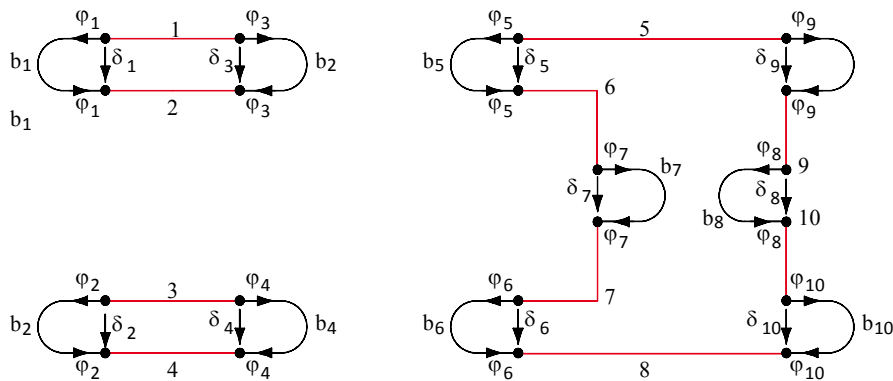


Figure 11: Connecting network in branch-/node-representation.

3.2.1. Graph

A graph $\Gamma = (N, E, \Phi)$ is defined as a set N of $|N|$ vertices v_k (*nodes*) that are connected by a set E of $|E|$ edges b_k (*branches*). The way the graph is connected is

given by the directed *incidence mapping* $\Phi: E \rightarrow N \times N$. Vertices connected by an edge are said to be *adjacent*. An example of a graph with its sets of vertices, edges and incidence mapping is given in **Fig. 12**.

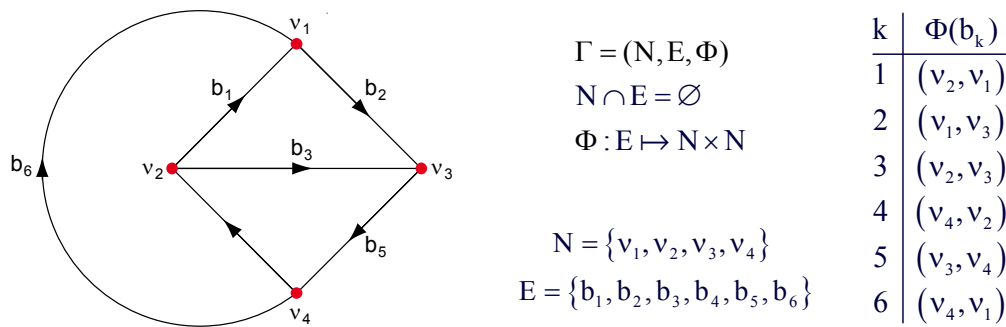


Figure 12: Example graph Γ with 6 branches, 4 nodes and incidence mapping.

3.2.2. Path, loops, components

To obtain a topological circuit description, we need to formulate nodal and loop equations. Given a lumped network graph, a loop l_ζ is any closed connected path $p_\zeta = \{v_{\zeta 1}, b_{\zeta 1}, v_{\zeta 2}, b_{\zeta 2}, \dots, v_{\zeta 1}\}$ among the graph's nodes and branches for which each included branch is traversed only once and each node encountered connects exactly two included branches. In **Fig. 12** an example for three loops $l_1 \dots l_3$ is given as well as the set of branches forming the loop with an incidence mapping describing the orientation of the loop with respect to the orientation of the branches (**Fig. 13**).

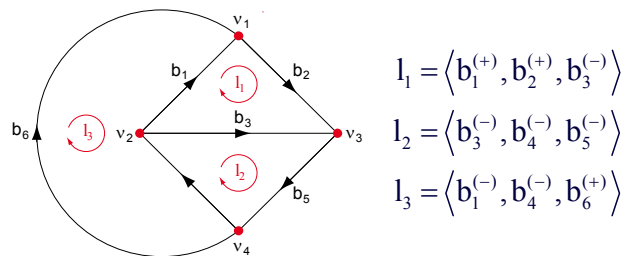


Figure 13: Example of loops in a graph Γ . The (+) or (-) specify the direction of the incidence.

To define nodal and loop incidence matrices and to take a closer look at their rank, the definitions of a connected graph and its components are needed: A lumped network graph is said to be connected if there exists at least one path P among its branches (disregarding the branches' orientations) between any pair of network nodes. A subgraph is a subset of the original set of graph branches along with their corresponding nodes. A maximally connected subgraph of a non-connected graph Γ is called a component C of Γ (**Fig. 14**).

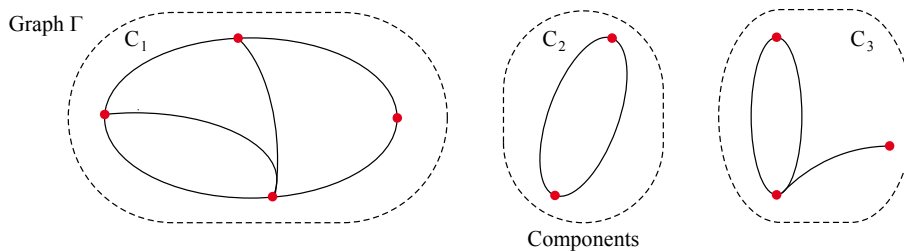
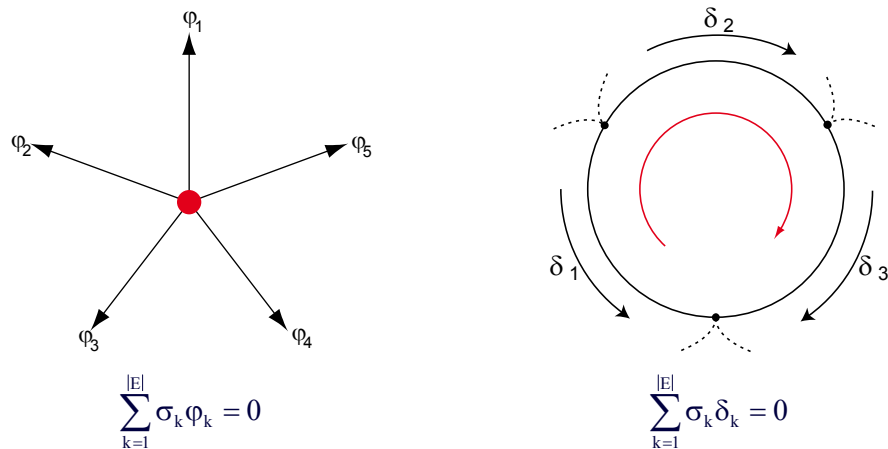


Figure 14: Example of a graph Γ consisting of 3 components C_1, C_2 and C_3 .

3.3. Nodal and loop incidence matrix

With the definitions of the previous section we can now formulate the *cutset law* for *through* and the *mesh law* for *across* quantities, in electrical domain *Kirchhoff's current law* and *Kirchhoff's voltage law*, which are illustrated in **Fig. 15**.



$\sigma_k \in [-1, 0, +1]$ according to the incidence $\Phi(b_k)$ of branch b_k with the node/loop.

Figure 15: Formulation of nodal equations for *through* and loop equations for *across* quantities.

Setting up the *cutset law* for each independent node v_p ($p \in |N| - |C|$, where $|C|$ is the number of the graph's components) yields a homogeneous matrix equation:

$$\sum_k \sigma_{p,k} \Phi_{p,k} = 0 \Leftrightarrow \mathbf{A}\boldsymbol{\varphi} = \mathbf{0} \quad (8)$$

with \mathbf{A} as nodal incidence matrix and $\boldsymbol{\varphi} = (\varphi_1, \dots, \varphi_{|E|})^T$ as vector of the *through* quantities. Accordingly the *mesh equations* can be formulated for each independent loop l_q in a similar way, *i.e.*:

$$\sum_k \sigma_{q,k} \Phi_{q,k} = 0 \Leftrightarrow \mathbf{B}\boldsymbol{\delta} = \mathbf{0} \quad (9)$$

with \mathbf{B} as loop incidence matrix and $\boldsymbol{\delta} = (\delta_1, \dots, \delta_{|E|})^T$ as vector of the *across* quantities.

An example of a graph's nodal and loop incidence matrices is given in **Fig. 16**.

Combining the *cutset law* / nodal equations (electrical: *Kirchhoff's Current Law* = KCL) and the *mesh law* / loop equations (electrical: *Kirchhoff's voltage law* = KVL) in a single matrix equation yields:

$$\underbrace{\begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} \end{bmatrix}}_{\mathbf{T}} \cdot \underbrace{\begin{bmatrix} \boldsymbol{\varphi} \\ \boldsymbol{\delta} \end{bmatrix}}_{\mathbf{a}} = \mathbf{0} \quad (10)$$

or written in a more compact form as a homogeneous system of linear equations:

$$\mathbf{T}\mathbf{a} = \mathbf{0} \quad (11)$$

Solving (11) yields a kernel $\mathbf{K} = \ker \mathbf{T} = \left\{ \mathbf{a} \in \mathbb{R}^{2|E|} \mid \mathbf{T}\mathbf{a} = \mathbf{0} \right\}$ whose dimension $\dim(\mathbf{K}) = \text{def}(\mathbf{T}) = 2|E| - |E| = |E|$ equals the number of branches $|E|$ and whose

solution is $\mathbf{K} = \text{span}(\mathbf{S}) = \left\{ \mathbf{a} \in \mathbb{R}^{2|E|} \mid \mathbf{a} = \mathbf{S} \mathbf{a}' \right\}$ with $\mathbf{S} \in \mathbb{R}^{2|E| \times |E|}$, $\mathbf{a}' \in \mathbb{R}^{|E|}$. This raises the interesting questions, if it is easily possible to find the kernel and, furthermore, if there is an interpretation for the result, especially for the vector \mathbf{a}' ?

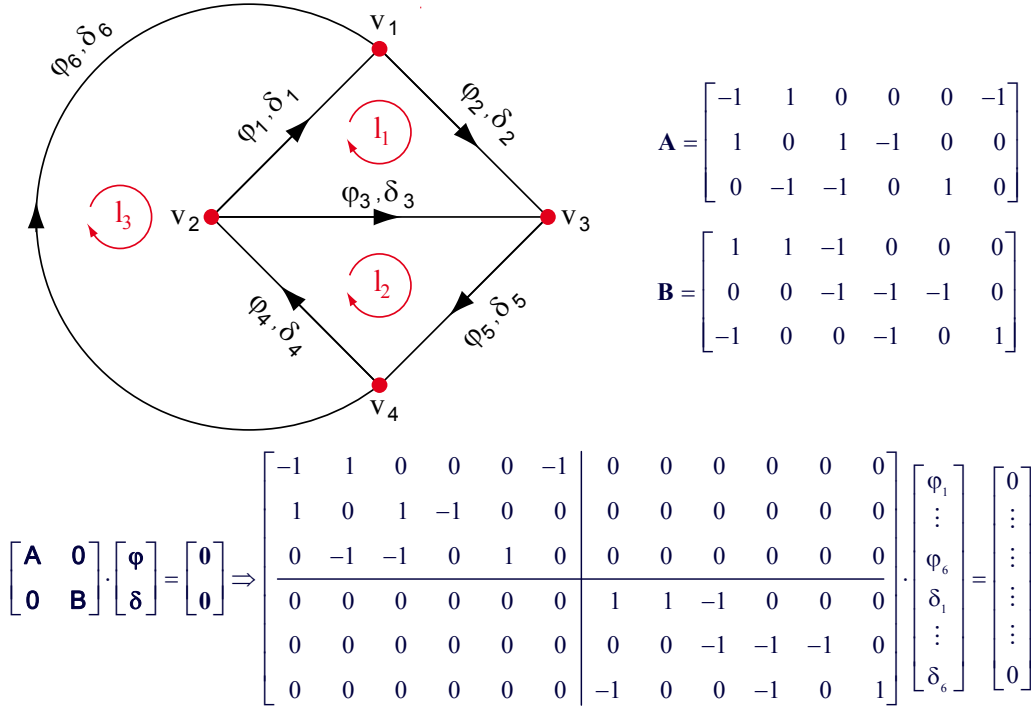


Figure 16: Example for formulation of nodal equations for through quantities and loop equations for across quantities using nodal and loop incidence matrices.

Definition: Exactness

A pair $(\mathbf{A}, \mathbf{A}')$ of matrices $\mathbf{A} \in \mathbb{R}^{n \times k}$, $\mathbf{A}' \in \mathbb{R}^{k \times m}$ is called exact if and only if:

$$\mathbf{A} \cdot \mathbf{A}' = \mathbf{0} \quad \text{and} \quad \text{rank}(\mathbf{A}) + \text{rank}(\mathbf{A}') = k \quad (12)$$

Theorem: The pair $(\mathbf{A}, \mathbf{B}^T)$ of incidence matrices is exact, i.e.

$$\mathbf{A} \cdot \mathbf{B}^T = \mathbf{0} \quad \text{and} \quad \text{rank}(\mathbf{A}) + \text{rank}(\mathbf{B}^T) = |E| \quad (13)$$

Hence, the rows of the nodal incidence matrix are orthogonal to the columns of the loop incidence matrix and vice versa.

Proof: The proof of the exactness of $(\mathbf{A}, \mathbf{B}^T)$ is not very complicated, because there are only four non-trivial cases (Fig. 17) of corresponding entries in the matrices \mathbf{A} and \mathbf{B} , but the resulting product equals zero in all of those cases.

Using the exactness of $(\mathbf{A}, \mathbf{B}^T)$ the general solution for the cutset (13) and mesh equations (24) – i.e. KCL and KVL equations – can be written as linear combination of the columns of the transposed incidence matrix of the opposite equation system:

$$\boldsymbol{\varphi} = \mathbf{B}^T \mathbf{j}, \quad \mathbf{j} \in \mathbb{R}^{|E|-|N|+1} \quad (14)$$

$$\boldsymbol{\delta} = \mathbf{A}^T \mathbf{v}, \quad \mathbf{v} \in \mathbb{R}^{|N|-1} \quad (15)$$

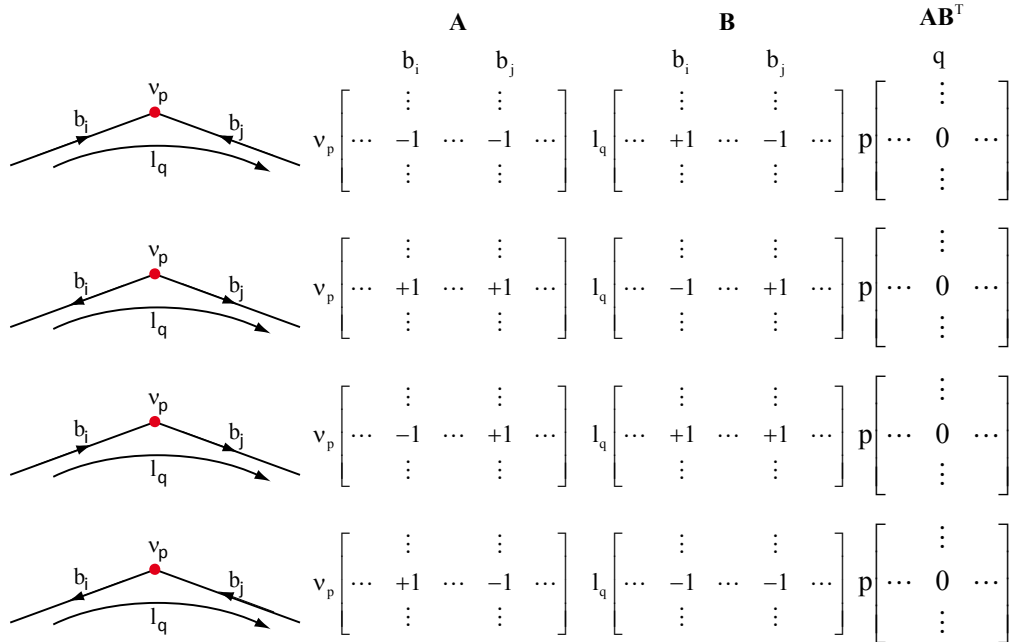


Figure 17: Proof of exactness condition for nodal and transposed loop incidence matrix based on 4 non-trivial cases.

Hence, the kernel of \mathbf{A} is formed by the column vectors of \mathbf{B}^T and vice versa the kernel of \mathbf{B} is formed by the column vectors of \mathbf{A}^T . The resulting full general solution of the *cutset* and *mesh law* is represented in Formula (16):

$$\begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{\varphi} \\ \boldsymbol{\delta} \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{B}^T \mathbf{j} \\ \mathbf{A}^T \mathbf{v} \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} \end{bmatrix}}_{\mathbf{T}} \cdot \underbrace{\begin{bmatrix} \mathbf{B}^T & \mathbf{0} \\ \mathbf{0} & \mathbf{A}^T \end{bmatrix}}_{\mathbf{S}} \cdot \underbrace{\begin{bmatrix} \mathbf{j} \\ \mathbf{v} \end{bmatrix}}_{\mathbf{a}'} = \mathbf{0} \quad (16)$$

It is interesting that there is also a technical interpretation of that result. As each branch voltage fulfilling KVL can be expressed by a linear combination of those columns of the transposed nodal incidence matrix \mathbf{A} , which belong to the incident nodes of the referring branch, the linear factor has to be of dimension voltage. Combining the “*node*” with a “*voltage*” yields a “*node voltage*” or a “*node potential*”, and indeed: assigning a potential to each node every branch voltage can be expressed by a difference of the node voltages of the respective branch that is connecting them. An example is given in **Fig. 18**.

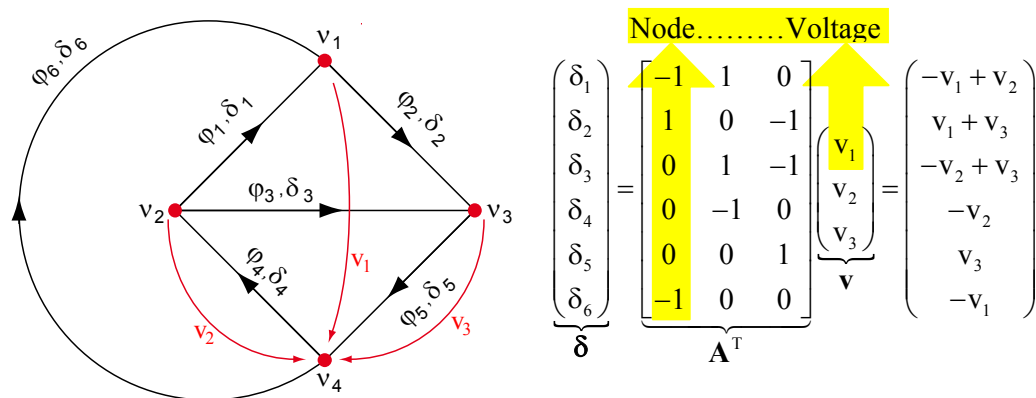


Figure 18: Interpretation of the kernel of KVL as node voltages or node potentials.

A similar interpretation can be found for the general solution of KCL (13) illustrated in Fig. 19: As each column of \mathbf{B}^T solves KCL, hence, the general solution is formed by a linear combination of all columns of \mathbf{B}^T . As \mathbf{B} is of no physical dimension and the general solution yields the branch currents the linear factors can be identified as “currents” assigned to each respective “loop”, i.e. “loop currents” or “mesh currents”. In summary: if all loop currents are known, each branch current can be expressed by a linear combination of loop currents, and the loop currents themselves fulfill KCL.

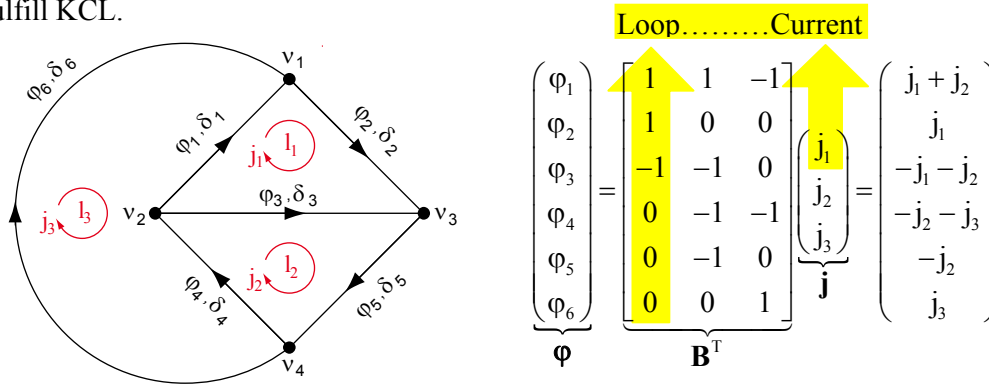


Figure 19: Interpretation of the kernel of KCL as loop currents or mesh currents.

As a consequence circuit analysis and equation formulation can be significantly reduced if either node voltages or loop currents are used as variables, i.e. KVL can be saved by using node voltages, and KCL by using loop currents only.

In the following section nodal (NA) and Modified Nodal Analysis (MNA) will be introduced that allow for an easy automated equation setup using node voltages and some selected currents as variables only.

3.4. General modified nodal analysis (MNA)

To derive a general approach for MNA we will separate the element relation vector into two parts, the first (upper one) that can be explicitly solved for *through* quantities (currents), and the second (lower part) for those relations that cannot be explicitly solved for certain *through* quantities (29).

$$\mathbf{f}(\boldsymbol{\varphi}, \boldsymbol{\delta}) = \mathbf{0} \Leftrightarrow \begin{pmatrix} \mathbf{f}_1(\boldsymbol{\varphi}_2, \boldsymbol{\delta}) - \boldsymbol{\varphi}_1 \\ \mathbf{f}_2(\boldsymbol{\varphi}_2, \boldsymbol{\delta}) \end{pmatrix} = \mathbf{0} \tag{17}$$

The first part is denoted by $\boldsymbol{\varphi}_1 = \mathbf{f}_1(\boldsymbol{\varphi}_2, \boldsymbol{\delta})$, meaning that the element relations \mathbf{f}_1 are functions of the *across* variables $\boldsymbol{\delta}$ (later expressed in node potentials) and a subset of *through* quantities $\boldsymbol{\varphi}_2$ (denoted by index "2") that can be explicitly solved for the subset of *through* quantities $\boldsymbol{\varphi}_1$ (denoted by index "1"). The implicit functions \mathbf{f}_2 of the second part denote the element relations of all those elements that cannot be explicitly solved for a *through* quantity, thus it is an implicit function of the *across* variables $\boldsymbol{\delta}$ (later expressed in node potentials) and the remaining *through* quantities $\boldsymbol{\varphi}_2$ (since all others are already included in \mathbf{f}_1).

As a consequence the circuit or system can be divided into two parts, i.e. each branch can be classified belonging to part one ("1") or part two ("2"), and hence branches, quantities and the nodal incidence matrix can also be structured in the same manner:

$$\mathbf{E} = [\mathbf{E}_1 \quad \mathbf{E}_2] \quad \boldsymbol{\varphi} = [\boldsymbol{\varphi}_1 \quad \boldsymbol{\varphi}_2] \quad \boldsymbol{\delta} = [\boldsymbol{\delta}_1 \quad \boldsymbol{\delta}_2] \quad \mathbf{A} = [\mathbf{A}_1 \quad \mathbf{A}_2] \tag{18}$$

As KCL (8) is still valid for the overall network, the following holds:

$$\mathbf{A}\boldsymbol{\varphi} = \mathbf{0} \Rightarrow \begin{bmatrix} \mathbf{A}_1 & \mathbf{A}_2 \end{bmatrix} \begin{pmatrix} \boldsymbol{\varphi}_1 \\ \boldsymbol{\varphi}_2 \end{pmatrix} = \mathbf{0} \Leftrightarrow \mathbf{A}_1\boldsymbol{\varphi}_1 = -\mathbf{A}_2\boldsymbol{\varphi}_2 \quad (19)$$

With (17) and (19) as well as (15) we obtain:

$$\begin{aligned} & \begin{bmatrix} \mathbf{A}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \cdot \begin{pmatrix} \mathbf{f}_1(\boldsymbol{\varphi}_2, \boldsymbol{\delta}) - \boldsymbol{\varphi}_1 \\ \mathbf{f}_2(\boldsymbol{\varphi}_2, \boldsymbol{\delta}) \end{pmatrix} \Big|_{\boldsymbol{\delta} = \mathbf{A}^T \mathbf{v}} = \mathbf{0} \\ & \Rightarrow \begin{pmatrix} \mathbf{A}_1 \mathbf{f}_1(\boldsymbol{\varphi}_2, \mathbf{A}^T \mathbf{v}) - \boldsymbol{\varphi}_1 \\ \mathbf{f}_2(\boldsymbol{\varphi}_2, \mathbf{A}^T \mathbf{v}) \end{pmatrix} = \mathbf{0} \\ & \Rightarrow \begin{pmatrix} \mathbf{A}_1 \mathbf{f}_1(\boldsymbol{\varphi}_2, \mathbf{A}^T \mathbf{v}) + \mathbf{A}_2 \boldsymbol{\varphi}_2 \\ \mathbf{f}_2(\boldsymbol{\varphi}_2, \mathbf{A}^T \mathbf{v}) \end{pmatrix} = \mathbf{0} \end{aligned} \quad (20)$$

Although **Error! Reference source not found.** may look complicated, MNA equation setup can be done very efficiently in an automated way by an element-wise processing from a netlist. Taking a closer look at **Error! Reference source not found.** two cases can be identified, illustrated in Fig. 20.

$$\text{Case 1) } b_i \in \mathbf{E}_1 \Leftrightarrow i \in [1, \dots, |\mathbf{E}| - |\mathbf{E}_2|]$$

$$\Rightarrow \varphi_i = \mathbf{f}_{1,i}(\boldsymbol{\varphi}_2, \boldsymbol{\delta})$$

$$\text{Case 2) } b_i \in \mathbf{E}_2 \Leftrightarrow i \in [|\mathbf{E}| - |\mathbf{E}_2| + 1, \dots, |\mathbf{E}|]$$

$$\Rightarrow \varphi_i = \mathbf{f}_{2, i-|\mathbf{E}+|\mathbf{E}_2|}(\boldsymbol{\varphi}_2, \boldsymbol{\delta}) = 0$$

$$\begin{pmatrix} \mathbf{A}_1 \mathbf{f}_1(\boldsymbol{\varphi}_2, \mathbf{A}^T \mathbf{v}) + \mathbf{A}_2 \boldsymbol{\varphi}_2 \\ \mathbf{f}_2(\boldsymbol{\varphi}_2, \mathbf{A}^T \mathbf{v}) \end{pmatrix} = \mathbf{0}$$

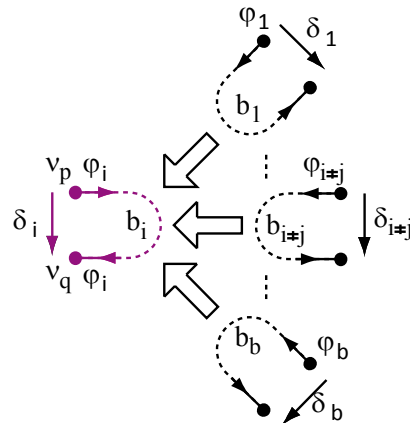


Figure 20: Element-wise setup of MNA equations.

If an element belongs to the first class of element relations, *i.e.* explicitly solvable for a *through* quantity, case “1” can be applied yielding the following MNA contribution:

$$\mathbf{A}_1 \mathbf{f}_1(\boldsymbol{\varphi}_2, \mathbf{A}^T \mathbf{v}) = \begin{matrix} & i \\ p_1 & \begin{bmatrix} \cdots & +1 & \cdots \\ & \vdots & \\ p_2 & \cdots & -1 & \cdots \end{bmatrix} \cdot \begin{pmatrix} \vdots \\ \mathbf{f}_{1,i}(\boldsymbol{\varphi}_2, \mathbf{A}^T \mathbf{v}) \\ \vdots \end{pmatrix} & i \\ & p_2 \end{matrix} = \begin{pmatrix} \vdots \\ +\mathbf{f}_{1,i}(\boldsymbol{\varphi}_2, \mathbf{A}^T \mathbf{v}) \\ \vdots \\ -\mathbf{f}_{1,i}(\boldsymbol{\varphi}_2, \mathbf{A}^T \mathbf{v}) \\ \vdots \end{pmatrix} \quad (21)$$

As it could be easily seen this is just the value of the *through* quantity (current) at the respective connected node according to the direction of the branch. For an element with index i belonging to the second class “2”, *i.e.* its element relation is not explicitly solvable for the *through* quantity, an additional equation (23) as well as the variable of the *through* quantity (22) has to be added to the MNA equations:

$$\mathbf{A}_2 \boldsymbol{\varphi}_2 = \begin{matrix} & & i \cdot |E| + |E|_2 \\ & & \vdots \\ p_1 & \cdots & +1 & \cdots \\ & & \vdots \\ p_2 & \cdots & -1 & \cdots \\ & & \vdots \end{matrix} \cdot \begin{pmatrix} \vdots \\ \boldsymbol{\varphi}_i \\ \vdots \end{pmatrix} = \begin{matrix} & & & & i \cdot |E| + |E|_2 \\ & & & & \vdots \\ p_1 & & & & +\boldsymbol{\varphi}_i \\ & & & & \vdots \\ p_2 & & & & -\boldsymbol{\varphi}_i \\ & & & & \vdots \end{matrix} \quad (22)$$

$$\mathbf{f}_2(\boldsymbol{\varphi}_2, \mathbf{A}^T \mathbf{v}) = i \cdot |E|_2 \left(\begin{matrix} \vdots \\ \mathbf{f}_{2, i \cdot |E|_2}(\boldsymbol{\varphi}_2, \mathbf{A}^T \mathbf{v}) \\ \vdots \end{matrix} \right) \quad (23)$$

For the respective node equations the contributions of the *through* quantities are added and as the *through* quantity has been introduced as new variable to the system an additional equation has to be provided, including just the implicit element relation of the respective element.

For linear element relations generic element patterns can be derived that leave the MNA-typical stamps in (20) yielding a linear matrix equation system [see Chapter 2] In the next section an example for setting up the equations of a nonlinear dynamic circuit is given.

3.4.1. Example: MNA equations of a common emitter amplifier

Fig. 21 shows a circuit of a simple common emitter amplifier. The related element relations are given in **Fig. 22**. To keep the equations not too complicated the BJT transistor is modeled by an ideal BE-diode and a linear current-controlled current source.

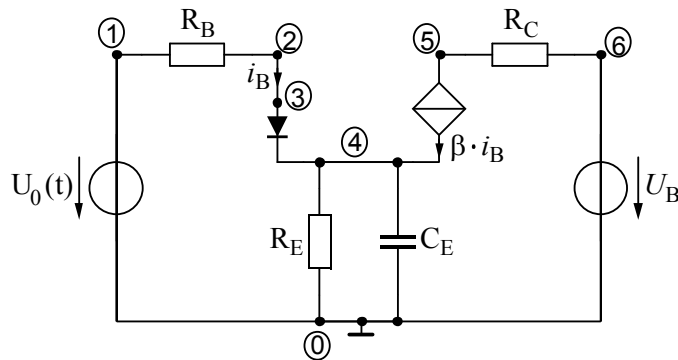


Figure 21: Circuit of a simple CE-amplifier.

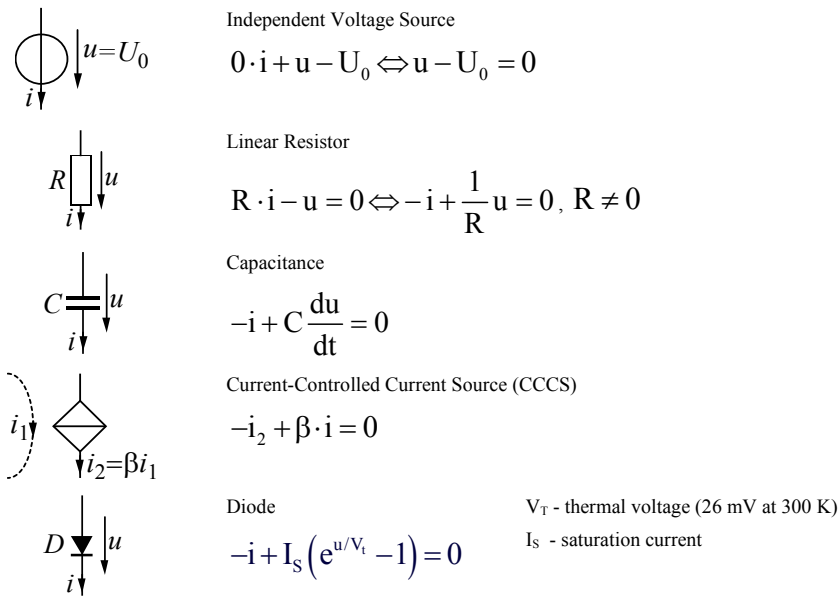


Figure 22: Element relations for the elements of the CE-amplifier circuit.

Fig. 23 then displays the graph of the circuit already separated according to element relations. There are 3 branches having a non-admittance representation, *i.e.* belonging to class “2”, hence the MNA equation system will have 3 more additional equations than the number of nodes.

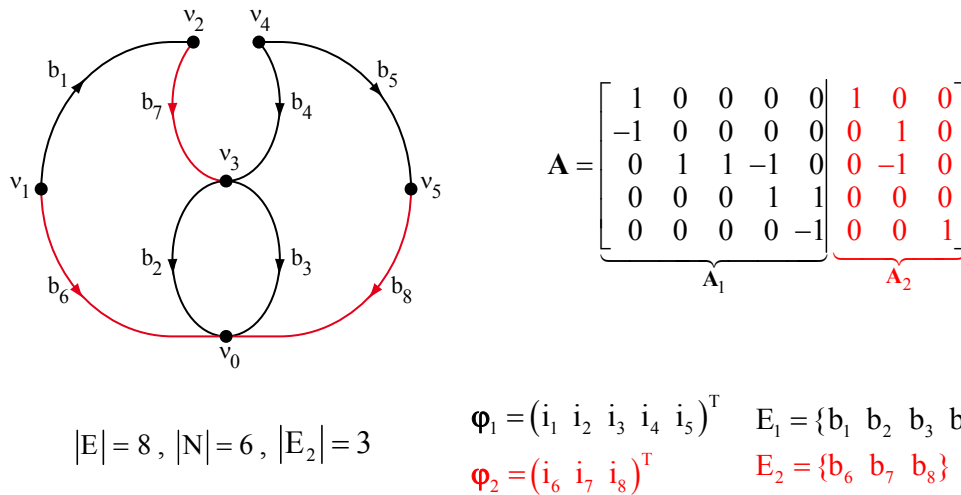


Figure 23: Graph of CE-amplifier with $|N| = 6$ nodes and $|E| = 8$ branches, whereas $|E_2| = 3$ branches are class “2” branches (left) and corresponding partitioned nodal incidence matrix (right).

The following Equations show the element-relation vectors of the amplifier for class “1” $\mathbf{f}_1(\boldsymbol{\varphi}_1, \boldsymbol{\delta})$ and class “2” $\mathbf{f}_2(\boldsymbol{\varphi}_2, \boldsymbol{\delta})$ or expressed in node voltages $\mathbf{f}_2(\boldsymbol{\varphi}_2, \mathbf{A}^T \mathbf{v})$:

$$\boldsymbol{\varphi}_1 = \mathbf{f}_1(\boldsymbol{\varphi}_2, \boldsymbol{\delta}) = \begin{pmatrix} \frac{1}{R_B} u_1 \\ \frac{1}{R_E} u_2 \\ C_E \dot{u}_3 \\ \beta i_7 \\ \frac{1}{R_C} u_5 \end{pmatrix} \Rightarrow \mathbf{f}_1(\boldsymbol{\varphi}_2, \mathbf{A}^T \mathbf{v}) = \begin{pmatrix} \frac{1}{R_B} (v_1 - v_2) \\ \frac{1}{R_E} v_3 \\ C_E \dot{v}_3 \\ \beta i_7 \\ \frac{1}{R_C} (v_4 - v_5) \end{pmatrix} \quad (24)$$

$$\mathbf{f}_2(\boldsymbol{\varphi}_2, \boldsymbol{\delta}) = \begin{pmatrix} u_6 - U_0(t) \\ I_S (e^{u_7/V_t} - 1) - i_7 \\ u_8 - U_B \end{pmatrix} = 0 \quad (25)$$

$$\Rightarrow \mathbf{f}_2(\boldsymbol{\varphi}_2, \mathbf{A}^T \mathbf{v}) = \begin{pmatrix} v_1 - U_0(t) \\ I_S (e^{(v_2 - v_3)/V_t} - 1) - i_7 \\ v_5 - U_B \end{pmatrix} = 0$$

Finally, **Fig. 24** contains the complete MNA system of equations with class “1” and class “2” relations highlighted as in **Fig. 20**.

$$\begin{array}{l} R_B \\ R_E \\ C_E \\ \text{CCCS} \\ R_C \\ U_0 \\ \text{Diode} \\ U_B \end{array} \left[\begin{array}{cc} +\frac{1}{R_B} (v_1 - v_2) & +i_6 \\ -\frac{1}{R_B} (v_1 - v_2) & +i_7 \\ +\frac{1}{R_E} v_3 + C_E \dot{v}_3 - \beta i_7 & \\ +\frac{1}{R_C} (v_4 - v_5) + \beta i_7 & -i_7 \\ -\frac{1}{R_C} (v_4 - v_5) & +i_8 \\ v_1 - U_0(t) & \\ +I_S (e^{(v_2 - v_3)/V_t} - 1) - i_7 & \\ v_5 - U_B & \end{array} \right] = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Figure 24: Complete MNA system of equations with highlighted class “1” and class “2” relations and order of the branch elements with respect to the branch enumeration on the left.

3.5. Automatic equation setup by symbolic analysis tool *Analog Insydes*

Now it will be outlined how the equations of the Common Emitter (CE) amplifier circuit will be automatically set up by *Analog Insydes*. The first step is to generate a netlist of the circuit which can be automatically done from schematics’ entries, e.g. *Cadence DFII* or *PSpice* Schematics editor. The generation of the *Analog Insydes* netlist is either done by a special netlister (*Cadence DFII*) or by *Analog Insydes*’ `ReadNetlist` command. It should be noted that for later symbolic approximation, the netlist should contain also numerical information about operating points as well as

model-card parameters. That's why *Analog Insydes'* netlister or netlist parsers also read in numerical simulation data like operating point information. **Fig. 25** again shows the circuit of the CE amplifier with node potentials marked. Note that for the current-controlled current source an additional short-circuit branch is needed to sense the controlling current. Hence, an additional node has been inserted so that the sensing branch connects node "2" and "3".

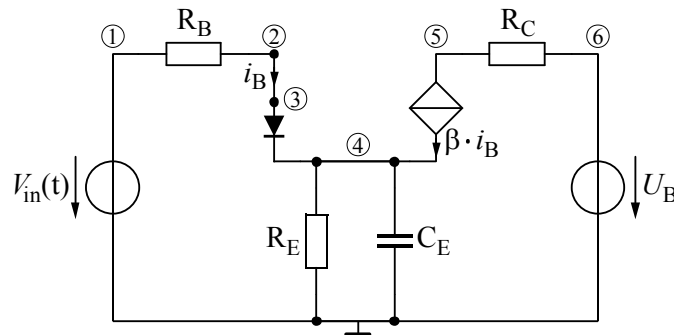


Figure 25: Equivalent circuit of a simple CE-amplifier prepared for *Analog Insydes*.

Fig. 26 displays the *Analog Insydes* netlist containing all information about the circuit, its elements, their symbols and values as well as the subcircuit definition for the diode. The way this subcircuit is defined will be explained in the following section on behavioral device models.

```
net = Circuit [
Netlist[{VIN, {1, 0}, Type → VoltageSource,
  Value → {AC → 1.~, DC | Transient → 5.~}, Symbolic → {AC | DC | Transient → VIN}},
{RB, {1, 2}, Type → Resistor, Value → 1000.~, Symbolic → RB},
{CCS, {2, 3, 5, 4}, Type → CCCSource, Value → 200.~, Symbolic → B},
{RE, {4, 0}, Type → Resistor, Value → 1000.~, Symbolic → RE},
{CE, {4, 0}, Type → Capacitor, Value → 10.*^-6, Symbolic → CE},
{RC, {5, 6}, Type → Resistor, Value → 1000.~, Symbolic → RC},
{VBATT, {6, 0}, Type → VoltageSource,
  Value → {AC → 0, DC | Transient → 10.~}, Symbolic → {DC | Transient → UB}},
{D1, {3 → "A", 4 → "C"}, Model → Diode, Selector → DCsimp, ID$ac → 0.0000227,
  VD$ac → 0.4230, REQ$ac → 2170.~, CAP$ac → 1.15*^-11, Is → 2.682*^-9, Vt → 0.026}
],
Model [
  Name → Diode,
  Selector → DCsimp,
  Ports → {"A", "C"},
  Parameters → {Is, Vt},
  Symbolic → {Is, Vt},
  Variables → {Current["A", "C"], Voltage["A", "C"]},
  Definition → Equations [
    Current["A", "C"] == Is (Exp[Voltage["A", "C"] / Vt] - 1)
  ]
]
]
```

Figure 26: *Analog Insydes'* netlist of the simple CE-amplifier.

3.6. Behavioral device models in *Analog Insydes*

Due to its origin *Analog Insydes* comes with a pre-defined device model library for analog electronic components. All device model implementations make use of the standard *Analog Insydes* modeling language in terms of a behavioral model description. This approach allows the modeling of nearly arbitrary element

characteristics by directly specifying the corresponding device equations, which in general may be a nonlinear DAE system. For this the *Analog Insydes* model definition is based on the port branch concept that was already introduced in Section 3.1.

Consider that in **Fig. 3** the anode and cathode pins of a diode are modeled as two connectors denoted by the identifiers “A” and “C”. At those ports (network nodes) different components can be interconnected. For the diode the current-voltage relation between those nodes is given by the device equation $I_D + I_S(e^{V_D/V_T} - 1) = 0$ for the branch voltage V_D and the branch current I_D , where I_S denotes the saturation current and V_T the thermal voltage.

Behavioral models can then be defined based on the port branches, where each unique pair of port identifiers (port1, port2) introduces a port branch between the model ports port1 and port2 with a positive reference direction from port1 to port2. The associated port variables in the behavioral model equations can be referred by means of special keywords, which are Voltage[port1, port2] and Current[port1, port2] for the electronic domain. For the diode the *Analog Insydes* format for the device equation (**Fig. 26** bottom part of the netlist) reads as:

$$\text{Current}[A,C] = I_D + I_S \left(e^{\text{Voltage}[A,C]/V_T} - 1 \right) \quad (26)$$

An alternative concept to the port branch concept could be the currents into the ports and the port voltages approach (as outlined in **Fig. 4** on the right side), but this method is not well suited for other analysis methods than MNA. Due to the fact that *Analog Insydes* supports different analysis methods the introduced port branch concept has been implemented instead.

3.7. Automated nonlinear time domain equation setup in MNA formulation

From the netlist of **Fig. 26** the circuit equations can be set up directly. **Fig. 27** shows the generated system of MNA equations which form a DAE system. The system is a *Mathematica* [13] list containing 3 entries: equations, vector of variables, and the design points that contain numerical values of all parameters of the equations.

$$\begin{aligned} & \left\{ \left\{ I\$VIN[t] + \frac{V\$1[t] - V\$2[t]}{RB} = 0, IC\$CCS[t] + \frac{-V\$1[t] + V\$2[t]}{RB} = 0, -IC\$CCS[t] + I\$AC\$D1[t] = 0, \right. \right. \\ & -\$ IC\$CCS[t] - I\$AC\$D1[t] + \frac{V\$4[t]}{RE} + CE V\$4'[t] = 0, \\ & \$ IC\$CCS[t] + \frac{V\$5[t] - V\$6[t]}{RC} = 0, I\$VBATT[t] + \frac{-V\$5[t] + V\$6[t]}{RC} = 0, \\ & V\$1[t] = VIN, V\$2[t] - V\$3[t] = 0, V\$6[t] = UB, I\$AC\$D1[t] = \left(-1 + e^{\frac{V\$3[t] - V\$4[t]}{Vt\$D1}} \right) I\$\$D1, \\ & \{V\$1[t], V\$2[t], V\$3[t], V\$4[t], V\$5[t], V\$6[t], I\$VIN[t], IC\$CCS[t], I\$VBATT[t], I\$AC\$D1[t]\}, \\ & \text{DesignPoint} \rightarrow \{VIN \rightarrow 5.^{\wedge}, RB \rightarrow 1000.^{\wedge}, \$ \rightarrow 200.^{\wedge}, RE \rightarrow 1000.^{\wedge}, \\ & CE \rightarrow 0.00001.^{\wedge}, RC \rightarrow 1000.^{\wedge}, UB \rightarrow 10.^{\wedge}, I\$\$D1 \rightarrow 2.682.^{\wedge} \cdot 10.^{-9}, Vt\$D1 \rightarrow 0.026.^{\wedge}\} \end{aligned}$$

Figure 27: System of MNA equations of the simple CE-amplifier generated by *Analog Insydes*.

Fig. 28 illustrates the relation between linear and nonlinear parts of the overall MNA system. The whole linear part of the circuit can be set up using matrix fill-in patterns. For the diode its current is introduced in the matrix, which means an additional column (here the last column of the matrix related to I_{D1}) but no additional row. Hence, the nonlinear element relation of the diode is added to the equations so that finally the same number of variables and equations is obtained.

$$\begin{pmatrix}
 \frac{1}{R_{IN}} & -\frac{1}{R_{IN}} & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
 \frac{1}{R_{IN}} & \frac{1}{R_{IN}} & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\
 0 & 0 & 0 & \frac{1}{R_E} + C_{ES} & 0 & 0 & 0 & 0 & -\beta & 0 \\
 0 & 0 & 0 & 0 & \frac{1}{RC} & -\frac{1}{RC} & 0 & \beta & 0 & 0 \\
 0 & 0 & 0 & 0 & -\frac{1}{RC} & \frac{1}{RC} & 0 & 0 & 1 & 0 \\
 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0
 \end{pmatrix}
 \cdot
 \begin{pmatrix}
 V_1 \\
 V_2 \\
 V_3 \\
 V_4 \\
 V_5 \\
 V_6 \\
 I_{V_{in}}(t) \\
 I_C\beta \\
 I_{UB} \\
 I_{DI}
 \end{pmatrix}
 =
 \begin{pmatrix}
 0 \\
 0 \\
 0 \\
 0 \\
 0 \\
 0 \\
 V_{in}(t) \\
 0 \\
 0 \\
 0
 \end{pmatrix}$$

$$I_{DI} = I_s \left(e^{\frac{V_1 - V_2}{V_T}} - 1 \right)$$

Figure 28: Illustration of the partitioning of the complete MNA system of the CE-amplifier into linear (top) and nonlinear (bottom) equations.

3.8. Example for a multiplier circuit using full SPICE-BJT models

Symbolic DAE systems of real-world circuits using fully simulator compatible device models (e.g. full Gummel-Poon BJT model) become rather large as motivated in Fig. 29.

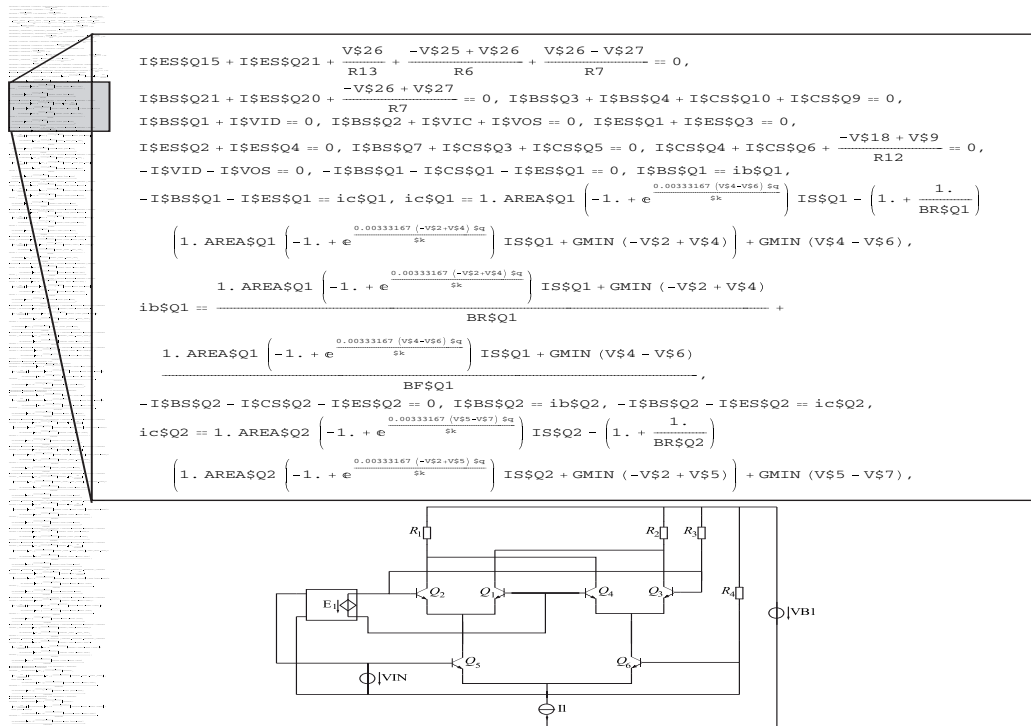


Figure 29: Impressions of a nonlinear DAE system for a 6-transistor multiplier circuit using full Gummel-Poon BJT models.

From this example, which is still rather small, it becomes obvious that the equations have to be simplified or approximated before they can be practically used *e.g.* as behavioral model. The next section will introduce symbolic approximation strategies for DAE systems and illustrate how reduced or simplified sets of equations can be generated that keep user defined accuracy compared to the original unsimplified equations.

4. Symbolic Approximation Strategies

4.1. Introduction

Practical application of symbolic analysis would have been rather limited without the application of symbolic approximation techniques. Indeed these techniques hold the key in modern symbolic circuit analysis. The concept “symbolic approximation” describes a whole class of mixed symbolic/numeric procedures for the completely automatic simplification of symbolic expressions using numerical evaluations and simulations to determine the approximation error. This is different from manual simplifications that are mainly based on qualitative considerations (*e.g.* $R_1 \ll R_2$) instead of quantitative ones. Automated symbolic approximation may yield more compact formulas satisfying a user-predefined error bound. Lots of research has been done and reported in this area resulting in three different categories of approximation strategies: *Simplification After Generation* (SAG) [14, 15], *Simplification During Generation* (SDG) [6, 16, 17] and *Simplification Before Generation* (SBG) [18-21].

One of the central prerequisites of the symbolic analysis flow presented in the next section was the development and implementation of efficient symbolic approximation algorithms, which impose no restrictions on the formulation of circuit equations, neither linear nor nonlinear, or on the set of circuit elements that may be used.

Equation-based approximation procedures own all these requested properties since they are already applied on the level of circuit equations before the solution is determined (SBG). The basic principle of equation-based approximation is to follow the methodology of a circuit designer who introduces his simplifications already when formulating equations. Thus the complexity of the problem and the mathematical effort to process and solve the system is reduced substantially.

4.2. Flow of equation-based approximation

Before going into more details the underlying principle of equation-based approximation is presented. **Fig. 30** shows a general flow of the introduced algorithm.

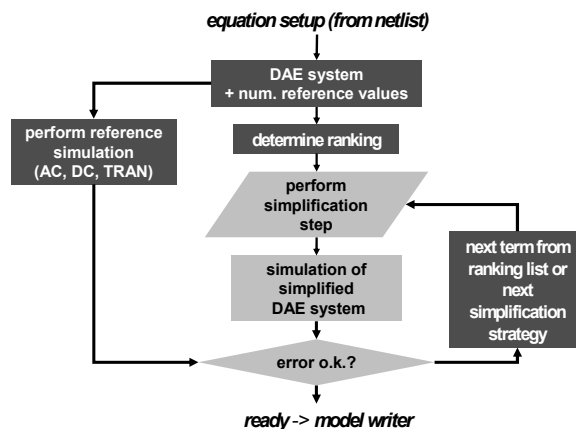


Figure 30: Flow of equation-based approximation.

Equation-based approximation starts with the system of symbolic linear or nonlinear equations and a list of corresponding numerical reference values describing specific design points. Based on those numerical reference values the system of symbolic equations is evaluated and numerically solved for the variable(s) of interest; *e.g.* the output voltage of a circuit. This may be done on linearized equations in the frequency domain (similar to the AC analysis of a circuit simulator like SPICE); on the static system of the nonlinear equations (DC or DC-transfer characteristics); or on the complete set of non-linear, dynamic DAEs in the time domain (transient analysis). In summary, the same analyses as those of numerical simulators can be performed. Then the corresponding results are compared to the output of the simulator in order to determine that all equations are correct and the symbolic model reduction can be performed next.

Starting with this full ranked system F of verified symbolic equations describing the system behavior the user chooses one or more numerical reference solutions f_i according to his interests as well as an appropriate error bound ε . Then the algorithm iteratively applies symbolic simplifications to the system and solves this simplified system numerically, whereas the model reduction consists of a sequence of different simplification steps (*e.g.* the deletion of entire expressions in a sum term), which are described below. Subsequently, the obtained solutions \tilde{f}_i are compared to the previously numerically computed reference solutions using an appropriate error norm: $\delta_i = \|f_i - \tilde{f}_i\|$. Thus, if some simplifications exceed the error bound, *i.e.* $\min(\delta_i) > \varepsilon$, the simplification is undone, otherwise it is accepted. These iteration steps are repeated until no more simplifications are possible without violating the error bound. Finally, the simplified symbolic system \tilde{F}_i is returned.

The order in which to simplify terms from the equation system is one of the crucial points: It is quite clear that those terms should be simplified first which have only a minor influence on the output behavior. Terms with large influence should not be removed at all. To achieve a maximum number of simplifications and to avoid unnecessary modifications an optimized order, the so called *ranking* is computed by a ranking algorithm predicting the influence on the error a modification step will cause. These ranking algorithms play a key role within the model-reduction process and depend on the circuit characteristics of interest. For example, in linear analysis magnitude, phase as well as pole and zero locations are of interest while in nonlinear analysis DC transfer, transient behavior, distortion, *etc.* are to be captured by the approximated system.

For linear systems a term ranking can be efficiently generated by application of the *Sherman-Morrison* formula [21-23], which allows calculating the deviation of one or more components of the solution vector with respect to perturbations within the system matrix. Another ranking criterion for linear systems is the calculation of eigenvalue sensitivities with respect to terms or entries in the system matrix [24].

In the nonlinear case ranking is much more difficult (more detailed descriptions can be found in [25, 26]). One of the methods is to perform only a single Newton step either in one point or on a set of points of the transient response starting from the original (reference) solution. The ranking of all possible simplification steps is computed by iteratively applying each simplification step to the original system and carrying out one single Newton step starting from the reference solution. The corresponding Newton deviations (corrections) derived from the first Newton steps

are employed for setting up the ranking order that gives an estimation if the corresponding simplification step is convenient or not.

According to the obtained ranking the equation-based approximation procedure iteratively applies single simplification steps and passes the manipulated system to the error checking routine, which calculates the accumulated numerical error of the simplification and compares it with the given error bound. In contrast to the results of the ranking - which only considers the error, a single simplification step causes on the original system - the accumulated error is the overall error when more simplification steps have been carried out according to the ranking list. If the error bound, *i.e.* the accumulated error, is exceeded the last simplification step is undone and the algorithm terminates returning the approximated system. Otherwise the next simplification is carried out according to the ranking list, followed by the error checking procedure as described before.

In order to reduce the computational effort caused by the error checking routine some simplification steps can often be combined to a cluster of simplifications having the same order of magnitude in the ranking. By this a whole cluster of simplifications can be applied at once in contrast to applying each of the simplifications separately.

4.3. Simplification strategies for nonlinear DAE systems

Currently, the following different simplification strategies are applied:

Algebraic Simplification, *i.e.* variable elimination or decoupling of independent blocks by exact algebraic transformations [27], *e.g.*:

$$\begin{aligned} x_1 &= y_1 + y_2 \\ x_2 &= y_2 + y_3 & \rightarrow & x_3 = y_1 \\ x_3 &= x_1 - x_2 + y_3 \end{aligned} \quad (27)$$

Branch Simplification, *i.e.* the deletion of those branches of piecewise-defined functions, which are not relevant during the simulation [27], *e.g.*:

$$y = x_1 + \begin{cases} x_1 x_2 & \text{if } x_2 > 1 \\ \frac{1}{2} x_1 (1 + x_2^2) & \text{else} \end{cases} \rightarrow y = x_1 + x_1 x_2 \quad (28)$$

for $x_2 > 1$ during the whole reference simulation

Switch Simplification, *i.e.* the consideration or neglect of terms, which are implemented suitably with respect to specific physical effects which may be taken into account (switch parameter $s=1$) or neglected (switch parameter $s=0$) [28] depending on use case parameters or simulation precision, *e.g.*:

$$M\ddot{x} = s_1 f(x) - s_2 \eta \dot{x} \rightarrow M\ddot{x} = f(x) \quad (29)$$

with the switches $s_1 = 1$, $s_2 = 0$ for turning on/off certain physical effects

Term Substitution, *i.e.* replacement of terms by their numerical mean value obtained during the reference simulation [26], *e.g.*:

$$\begin{aligned} x_3 &= R(x_1 + f(x_2)) \rightarrow x_3 = R(x_1 + 1.23) \\ &\text{for } \overline{f(x_2)} = 1.23 \end{aligned} \quad (30)$$

Term Deletion, *i.e.* the removal of terms from the system of equations [26], *e.g.*:

$$x_3 = \left(R_1 + \cancel{R_2} \right) \left(x_1 + \cancel{f(x_2)} \right) \rightarrow x_3 = R_1 x_1 \quad (31)$$

for $R_2 \ll R_1$ and $f(x_2) \ll x_1$

Within these definitions the item *term* describes all symbolic expressions that appear as summands in the equations and which can consist of expressions themselves again (term-in-term or hierarchical term deletion).

The application of algebraic and branch simplification methods do not influence the numeric solution of the DAE system, whereas switch simplification, term substitution, and term deletion are approximations, which lead to errors or deviations from the original solution. However, all simplifications can influence the numerical stability of the system of equations. For this reason dedicated methods have been developed [27] to monitor the numerical stability during the simplification process. The stability and other analytical properties of numerical DAE systems depend essentially on the differential index (see *e.g.* [29]). DAEs with an index zero or one can be solved easily with standard ODE methods. However, higher index problems (*i.e.*, DAEs with an index larger than one) constitute serious problems to the numerical solvers. It has been observed that the symbolic simplification of a DAE system may increase the system's index. As a consequence an index observer has been integrated into the simplification algorithm in order to make sure that the index is not increased by the simplification procedure.

The overall duration of approximation and behavioral model generation depends strongly on the equations' complexity as well as on the type and properties of simulations that have to be performed (AC, DC, transient). While the nonlinear dynamic approximation (DC and transient error criterion) of a 8 transistor operational amplifier (**Fig. 31**: 73 equations, 350 terms, 94 parameters) to a simplified system (**Fig. 34**: 6 equations, 24 terms, 21 parameters) takes about 13 minutes, computation time may increase to several hours for larger circuits, *e.g.* a μ A741 operational amplifier (26 transistors). On the other hand, linear symbolic approximation (transfer functions, poles and zeros) is performed in interactive computation time: For industrial-sized analog circuits, *e.g.* for up to 50 transistors, approximation times are a few minutes or less.

Example

In the following example it will be demonstrated that for real-world circuits the reduction can be enormous even without losing too much accuracy. **Fig. 31** shows an 8 transistor operational amplifier circuit. The task is to derive a behavioral model having a DC as well as a transient specification, *i.e.* that the DC error between original circuit and generated behavioral model should not exceed 0.25V for an large signal input range from -1V to 1V, and that the transient error for a square-wave signal should not exceed 1V (norm or maximum norm), see **Fig. 32**.

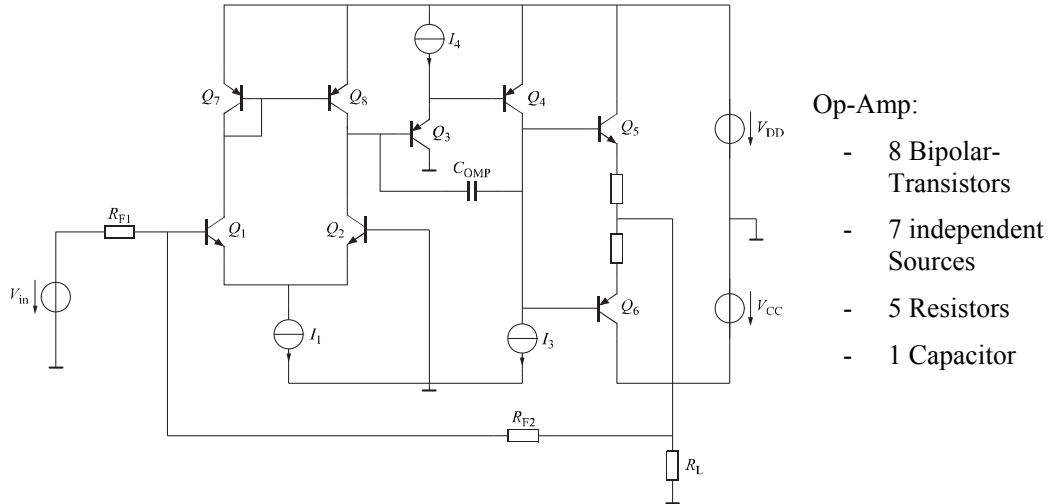
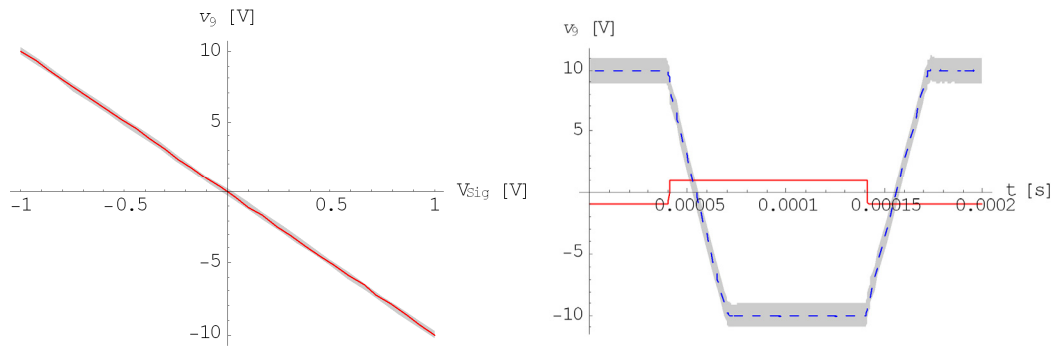


Figure 31: Operational amplifier circuit to demonstrate automated behavioral model generation.



DC-transfer specification

- error bound: $\epsilon_{DC} = 0.25V$
- error function: $E_{DC} = \|\bullet\|_{\infty}$

Transient-transfer specification

- error bound: $\epsilon_{Tran} = 1.0V$
- error function: $E_{Tran} = \|\bullet\|_{\infty}$

Figure 32: User-defined error specification for model generation: DC (left) and transient (right) error functions (input: continuous line, red; output: dashed line, blue; error bound: gray).

Setting up the equations of the full circuit – *i.e.* the SPICE simulator equivalent DAE system – yields 73 equations with 350 terms in 94 parameters. This complexity is illustrated in **Fig. 33**. The full printout would fill pages.

Original system: 73 equations, 350 terms, and 94 parameters.

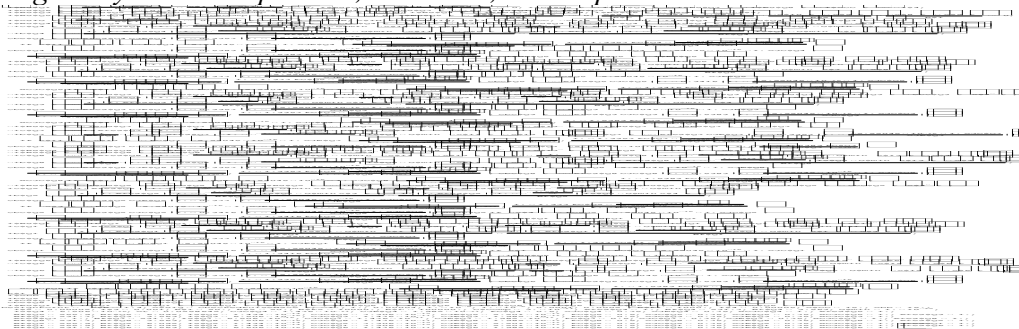


Figure 33: Illustration of the original DAE system of the op-amp circuit.

After application of the different simplification steps (workflow according Fig. 30 algebraic compression, branch simplification, algebraic compression, term deletion, algebraic compression) the DAE system displayed in Fig. 34 is obtained.

Symbolic Approximation: 73 → 6 equations, 350 → 24 terms and 94 → 21 parameters

$$\begin{aligned}
 I_1 R_{F1} - V_{Sig} - e^{-\frac{v_{j3}(t)}{Vt}} \text{Area}_{Q8} I_{S_{Q8}} R_{F1} - e^{\frac{v_j(t)}{Vt} - \frac{v_3(t)}{Vt}} \text{Area}_{Q1} I_{S_{Q1}} R_{F1} + v_j(t) + \frac{R_{F1} v_j(t)}{R_{F2}} - \frac{R_{F1} v_9(t)}{R_{F2}} - C_{Comp} R_{F1} v_j'(t) &= 0 \\
 -e^{-\frac{v_7(t)}{Vt} - \frac{v_8(t)}{Vt}} \text{Area}_{Q5} I_{S_{Q5}} + \frac{v_8(t)}{R_1} - \frac{v_9(t)}{R_1} &= 0 \\
 e^{-\frac{v_{j3}(t)}{Vt}} \text{Area}_{Q7} I_{S_{Q7}} - e^{\frac{v_j(t)}{Vt} - \frac{v_3(t)}{Vt}} \text{Area}_{Q1} I_{S_{Q1}} - 1.3357 \text{Area}_{Q8} C_{JE_{Q8}} v_{j3}'(t) &= 0 \\
 e^{-\frac{v_{j3}(t)}{Vt}} \text{Area}_{Q8} I_{S_{Q8}} - e^{-\frac{v_3(t)}{Vt}} \text{Area}_{Q2} I_{S_{Q2}} + C_{Comp} v_j'(t) &= 0 \\
 I_1 - e^{-\frac{v_{j3}(t)}{Vt}} \text{Area}_{Q8} I_{S_{Q8}} - e^{\frac{v_j(t)}{Vt} - \frac{v_3(t)}{Vt}} \text{Area}_{Q1} I_{S_{Q1}} - C_{Comp} v_j'(t) &= 0 \\
 e^{-\frac{v_7(t)}{Vt} - \frac{R_2 v_8(t) - R_1 v_9(t) - R_2 v_9(t)}{Vt R_1}} \text{Area}_{Q6} I_{S_{Q6}} - \frac{v_8(t)}{R_1} + \frac{v_9(t)}{R_1} &= 0
 \end{aligned}$$

Figure 34: Approximation result: simplified DAE system generated by *Analog Insydes*.

Taking a look at the comparison of simulation results in Fig. 35 it becomes obvious how powerful symbolic approximation is and that such dramatic simplifications can never be obtained by a manual procedure. Moreover, this result shows that - at least for well-designed circuits - a significant reduction can be performed without losing too much accuracy. For more critical circuits with more interaction between the elements and with larger influence of parasitics the remaining equations will be more complex due to more coupling and higher sensitivity with respect to the involved terms and components.

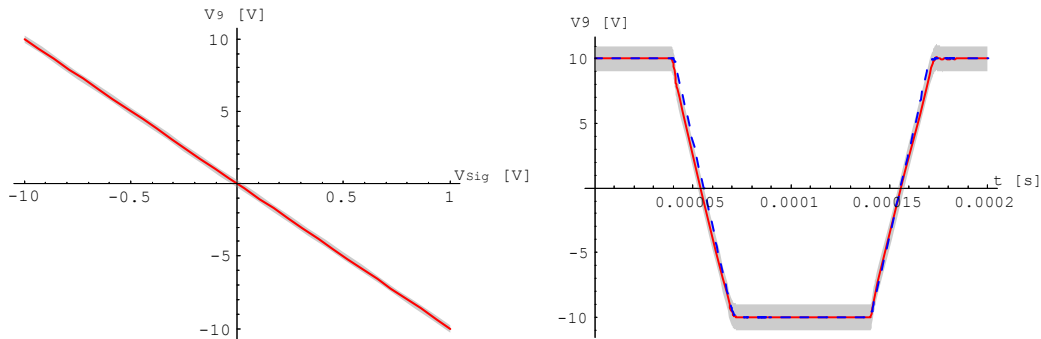


Figure 35: Comparison of the results, showing the original (continuous red line), the approximated system (dashed blue line) and the related error boundaries (gray).

4.4. Transfer of the methodology

Because of the general mathematical approach of the model processing procedures the methods originally developed for analog circuits can be transferred to the other application fields as well. *Analog Insydes'* capabilities for automatic generation of symbolic equations and their approximation can be applied to any system that is described as generalized Kirchhoff networks (*i.e.* by generalized Kirchhoff equations). Thanks to the analogies to electronic circuits (Section 3) all *through* and *across* quantities can be interpreted as “currents” and “voltages” without any changes of the program code. Thus, only some extensions of the model libraries were necessary to adapt the methods of automated equation setup and symbolic approximation to the analysis of other domains systems as *e.g.* hydraulics (*e.g.* gas-pipeline nets [30]) and mechatronics (*e.g.* acceleration sensor [31]).

For the handling of mechanical and mechatronical systems the *Fraunhofer* Institutes *ITWM* and *IIS/EAS* developed an *Analog Insydes* library, which contains linear and

some nonlinear beam elements as well as different force and displacements sources, providing symbolic models of micromechanical elements according to the introduced modeling approach for mechanical system elements (see Section 3.1).

4.5. Behavioral device models of generic multi-domain multi-port/-branch elements

To adapt this concept to generic multi-domain systems, the so far scalar-type electrical ports of *Analog Insydes* have been extended to the more general case of multidimensional vector-type *through* and *across* quantities. Considering each dimension of each related pair of quantities as a separate branch, a simple 3-dimensional mechanical component with two ports has already six accompanying model branches (e.g. see beam element in **Fig. 6**). To reduce failure and effort in *Analog Insydes* the vector-type description of multidimensional quantities is implemented.

Within an *Analog Insydes* netlist description this is achieved by adding a `NetlistAttributes` section, a new language object that is valid for the whole netlist object within which it has been defined. Thus simply the dimension of a corresponding netlist node has to be defined using the `NodeDimensions` keyword. The appropriate syntax for this is as follows:

```
NetlistAttributes[
  NodeDimensions -> {<node1> -> 6, ..., Default -> 1},
  NodePositions -> {<node1> -> {<x1>, <y1>, <z1>}, ...}
]
```

Here `<node1>` denotes the name of the mechanical node and `{<x1>, <y1>, <z1>}` is the corresponding numerical coordinate vector, respectively. The coordinate information specified by the `NodePositions` keyword is used for the automatic computation of the geometrical parameters of the mechanical component and with the `Default -> 1` setting all remaining nodes in the netlist description can be defined as being of scalar type without stating them explicitly, which is very useful when having multi-physics applications. This new language construct is used in the following examples.

5. Application Example for a Mechatronic System

As an example for a multi-physical system we consider an acceleration sensor [32] consisting of mechanical and electrical parts (**Fig. 36**). The sensor contains three parallel conducting plates, which form two serial capacities C_1 and C_2 . The central plate is versatile and can be moved out of its balanced position (central if $R_A = R_B$) resulting in a *Hooke's force* with constant K . In case of acceleration, the central plate is forced to leave its central position resulting in changes of the capacities between the electrical connectors $E1/E0$ and $E2/E0$. This yields a potential drop V_{out} for the central plate with respect to its potential in the idle state.

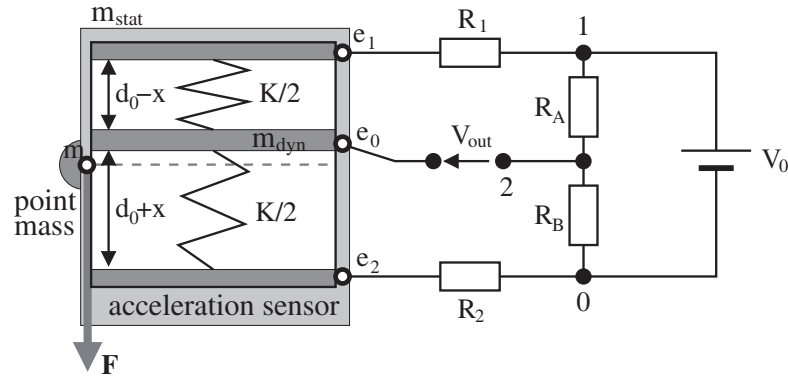


Figure 36: Acceleration sensor with simple circuitry.

This acceleration sensor has one mechanical and three electrical ports (the center that is connected to mass potential and each plate). The mechanical port has the vector variables displacement u and force F . Besides the accelerating external force F internal forces are acting on the system, resulting from electrostatics, *Hooke's law*, and damping, described by:

$$F_{\text{int}} = \frac{Q_1^2 - Q_2^2}{2A\epsilon_0} - Kx - D\dot{x} \quad (32)$$

Where Q_1 and Q_2 are the charges of the plates E_1 and E_2 , A is the plate area, ϵ_0 the dielectric constant, and D the damping constant. Acting along the axial direction of e the force results in an acceleration of the central plate whose mass is m_{dyn} :

$$F_{\text{dyn}} = m_{\text{dyn}} (\ddot{x} + e \cdot \ddot{u}) \quad (33)$$

Here, \mathbf{x} is the local displacement of the central plate from the idle position and u the global displacement of the acceleration sensor. Thus the forces acting on the static mass m_{stat} are the external force F and the internal force $F_{\text{int}} e$ yielding the equation of motion:

$$F - F_{\text{int}} e = m_{\text{stat}} \ddot{u} \quad (34)$$

For the capacities the charges Q_1 and Q_2 depend on the node voltages V_0 , V_1 and V_2 at the electrical connection ports E_0 , E_1 and E_2 :

$$Q_1 = C_1 (V_1 - V_0) \quad Q_2 = C_2 (V_2 - V_0) \quad (35)$$

where the capacitances C_1 and C_2 depend on the plate distances (idle distance d_0):

$$C_1 = \epsilon \frac{A}{d_0 - |x|} \quad C_2 = \epsilon \frac{A}{d_0 + |x|} \quad (36)$$

The branch currents from E_1 to E_0 and E_2 to E_0 are given by:

$$i_{k,0} = \dot{Q}_k, \quad k \in \{1, 2\} \quad (37)$$

Because of using six-dimensional port variables (3 for the displacements and 3 for the rotation angles), we add trivial angular equations with moment of inertia θ and torque M :

$$\ddot{\phi}_k = \theta M_k, \quad k \in \{1, 2, 3\} \quad (38)$$

```

Circuit[
  Netlist[
    {V0, {1, 0}, Symbolic → V0, Value → 1},
    {RA, {1, 2}, Symbolic → RA, Value → 1},
    {RB, {2, 0}, Symbolic → RB, Value → 1},
    {R1, {1, e1}, Symbolic → R1, Value → 1. × 108},
    {R2, {0, e2}, Symbolic → R2, Value → 1. × 108},
    {out, {2, e0}, Value → 0, Type → OpenCircuit},

    {as, {m → M, e0 → E0, e1 → E1, e2 → E2},
     Model → AccelerationSensor, DIRECTION → {0, 1., 0}},
    {mass, {m → M}, Model → Mass, M → 0.1},
    {force, {m → A}, Model → Source,
     FORCE → {0, If[t > 0.01, -1, 0], 0}, MOMENT → {0, 0, 0}},

    NetlistAttributes[
      NodeDimensions → {m → 6, Default → 1},
      NodePositions → {m → {0, 0, 0}}
    ]
  ]
]

```

Figure 37: Multi-physics netlist including acceleration sensor.

Changes of the orientation of the sensor are not considered in this example. The above equations are implemented as an *Analog Insydes* model called `AccelerationSensor`.

Fig. 37 shows its usage within the netlist description for the system of **Fig. 36**. The acceleration sensor `as` is oriented in $(0,1,0)$ direction. Its connections are specified by `<node> -> <port>` mappings. On the one hand the electrical ports `E0`, `E1` and `E2` are connected to the circuit at nodes `e0`, `e1` and `e2`; on the other hand the mechanical port `M` is connected to the point mass and the accelerating force at node `m`.

At this node a time-dependent force acts towards $(0,-1,0)$, accelerating the system starting at $t=10$ ms. The corresponding numerical values for all model parameters are listed in **Fig. 38**. All component parameters are instantiated by the postfix `$<comp>` where `<comp>` specifies the corresponding component name as given in the netlist. For all values which are not set within the netlist, default values are used instead.

<code>V0 → 1</code>	<code>DAMP\$as → 0.0485322</code>
<code>RA → 1</code>	<code>D0\$as → $\frac{1}{125\,000}$</code>
<code>RB → 1</code>	<code>THETA\$as → 0.0001</code>
<code>R1 → 1. × 10⁸</code>	<code>E1\$as → 0</code>
<code>R2 → 1. × 10⁸</code>	<code>E2\$as → 1.</code>
<code>M\$mass → 0.1</code>	<code>E3\$as → 0</code>
<code>THETA\$mass → 1.</code>	<code>F1\$force → 0</code>
<code>AREA\$as → 6.25 × 10⁻⁶</code>	<code>F2\$force → If[t > 0.01, -1, 0]</code>
<code>EPS0\$as → 8.85419 × 10⁻¹²</code>	<code>F3\$force → 0</code>
<code>K\$as → 60.086</code>	<code>M1\$force → 0</code>
<code>MSTAT\$as → 0.001927</code>	<code>M2\$force → 0</code>
<code>MDYN\$as → 9.8 × 10⁻⁶</code>	<code>M3\$force → 0</code>

Figure 38: Numerical values for all system parameters (given in SI units).

$$\begin{aligned}
i\$V0[t] + \frac{v\$1[t]}{R1} + \frac{v\$1[t]}{RA} - \frac{v\$2[t]}{RA} - \frac{v\$e1[t]}{R1} &== 0 \\
-\frac{v\$1[t]}{RA} + \frac{v\$2[t]}{RA} + \frac{v\$2[t]}{RB} &== 0 \\
-i\$E1E0\$as[t] - i\$E2E0\$as[t] &== 0 \\
i\$E1E0\$as[t] - \frac{v\$1[t]}{R1} + \frac{v\$e1[t]}{R1} &== 0 \\
i\$E2E0\$as[t] + \frac{v\$e2[t]}{R2} &== 0 \\
f\$M\$10\$as[t] + f\$M\$10\$mass[t] + f\$A\$10\$force[t] &== 0 \\
f\$A\$20\$force[t] + f\$M\$20\$mass[t] + f\$M\$20\$as[t] &== 0 \\
f\$M\$30\$mass[t] + f\$A\$30\$force[t] + f\$M\$30\$as[t] &== 0 \\
f\$M\$40\$mass[t] + f\$A\$40\$force[t] + f\$M\$40\$as[t] &== 0 \\
f\$M\$50\$as[t] + f\$A\$50\$force[t] + f\$M\$50\$mass[t] &== 0 \\
f\$M\$60\$as[t] + f\$M\$60\$mass[t] + f\$A\$60\$force[t] &== 0 \\
-V0 + v\$1[t] &== 0 \\
-v\$2[t] + v\$e0[t] + v\$out[t] &== 0 \\
-f\$M\$10\$mass[t] - M\$mass u\$m\$1''[t] &== 0 \\
-f\$M\$20\$mass[t] - M\$mass u\$m\$2''[t] &== 0 \\
-f\$M\$30\$mass[t] - M\$mass u\$m\$3''[t] &== 0 \\
-f\$M\$40\$mass[t] - THETA\$mass u\$m\$4''[t] &== 0 \\
-f\$M\$50\$mass[t] - THETA\$mass u\$m\$5''[t] &== 0 \\
-f\$M\$60\$mass[t] - THETA\$mass u\$m\$6''[t] &== 0 \\
c1\$as[t] - \frac{AREA\$as EPS0\$as}{DO\$as-x\$as[t]} &== 0 \\
c2\$as[t] - \frac{AREA\$as EPS0\$as}{DO\$as+x\$as[t]} &== 0 \\
q1\$as[t] + c1\$as[t] v\$e0[t] - c1\$as[t] v\$e1[t] &== 0 \\
q2\$as[t] + c2\$as[t] v\$e0[t] - c2\$as[t] v\$e2[t] &== 0 \\
0 == fint\$as[t] + \frac{q1\$as[t]^2}{2 AREA\$as EPS0\$as} + \frac{q2\$as[t]^2}{AREA\$as EPS0\$as} + K\$as x\$as[t] + & \\
+DAMP\$as x\$as'[t] & \\
i\$E1E0\$as[t] - q1\$as'[t] &== 0 \\
i\$E2E0\$as[t] - q2\$as'[t] &== 0 \\
0 == -fint\$as[t] + E1\$as MDYN\$as u\$m\$1''[t] + E3\$as MDYN\$as u\$m\$3''[t] + & \\
+ MDYN\$as x\$as''[t] + E2\$as MDYN\$as u\$m\$2''[t] & \\
f\$M\$10\$as[t] + E1\$as fint\$as[t] + MSTAT\$as u\$m\$1''[t] &== 0 \\
f\$M\$20\$as[t] + E2\$as fint\$as[t] + MSTAT\$as u\$m\$2''[t] &== 0 \\
f\$M\$30\$as[t] + E3\$as fint\$as[t] + MSTAT\$as u\$m\$3''[t] &== 0 \\
f\$M\$40\$as[t] + THETA\$as u\$m\$4''[t] &== 0 \\
f\$M\$50\$as[t] + THETA\$as u\$m\$5''[t] &== 0 \\
f\$M\$60\$as[t] + THETA\$as u\$m\$6''[t] &== 0 \\
f\$A\$10\$force[t] - F1\$force &== 0 \\
-F2\$force + f\$A\$20\$force[t] &== 0 \\
f\$A\$30\$force[t] - F3\$force &== 0 \\
f\$A\$40\$force[t] - M1\$force &== 0 \\
f\$A\$50\$force[t] - M2\$force &== 0 \\
f\$A\$60\$force[t] - M3\$force &== 0 \\
u\$m\$1[0] &== 0 \\
u\$m\$2[0] &== 0 \\
u\$m\$3[0] &== 0 \\
u\$m\$4[0] &== 0 \\
u\$m\$5[0] &== 0 \\
u\$m\$6[0] &== 0 \\
u\$m\$1'[0] &== 0 \\
u\$m\$2'[0] &== 0 \\
u\$m\$3'[0] &== 0 \\
u\$m\$4'[0] &== 0 \\
u\$m\$5'[0] &== 0 \\
u\$m\$6'[0] &== 0 \\
q1\$as[0] &== 0 \\
q2\$as[0] &== 0 \\
x\$as[0] &== 0 \\
x\$as'[0] &== 0
\end{aligned}$$

Figure 39: Automatically generated DAE system.

This notation is also used for the DAE system shown in **Fig. 39**, which has been generated automatically from the netlist description. It consists of 39 equations for 39 variables that are named, using a similar convention: *e.g.* the local displacement x of the central plate of as is named $x_{as}[t]$ and the third component of the displacement in node M is denoted by $u_{m3}[t]$. Additionally, there are 16 initial conditions for the 14 mechanical variables (location and rotation angle, velocity and angular speed, displacement and velocity of the central plate) as well as the charges on plates $E1$ and $E2$.

To perform the approximation as described in Section 4.2, the output variable V_{out} has been chosen as variable of interest and the maximum approximation error has been set to 20 mV (20%). For these settings, the highlighted expressions have been identified to be not mandatory relevant for the dynamics of V_{out} . Furthermore, automated exact algebraic manipulation finally leads to the reduced DAE system shown in **Fig. 40**, which consists of nothing but five equations only. Note, that the reduction to the y direction (only displacement variable u_{m2} is left) and the removal of all rotational degrees of freedom has been done in a completely automatic way. Additional approximations, *e.g.* removal of the expression $(MDYN_{as} x_{as}''[t])$ in the dynamics of the central plate, have also been applied automatically. Here, “ $MDYN_{as} x_{as}''[t]$ ” is an expression in *Mathematica* syntax. It is the product of the model parameter $MDYN_{as}$ (dynamic mass of acceleration sensor as) times the second derivative of the variable $x_{as}[t]$ (displacement x of the central plate of the acceleration sensor as).

Finally, **Fig. 41** illustrates a comparison of V_{out} for the original and the simplified DAE systems, considering the acceleration of the point mass by a force, starting at $t = 10$ ms. In the reduced model, the systems acceleration only depends on the point mass M_{mass} , neglecting the mass of the sensor ($M_{STAT}_{as} + MDYN_{as}$), which is less than two percent of the point mass. This finally results in the slightly increased absolute value for the stationary voltage V_{out} . The different dynamic behavior close to $t = 10$ ms is mainly due to neglecting the inertia of the central plate.

$$\begin{aligned}
 F2_{force} &== M_{mass} u_{m2}''[t] \\
 q1_{as}[t] &== \frac{AREA_{as} EPS0_{as} (RA V0 + (RA+RB) (v_{out}[t] + R1 q2_{as}'[t]))}{(RA+RB) (D0_{as} - x_{as}[t])} \\
 q2_{as}[t] &== \frac{AREA_{as} EPS0_{as} (-RB V0 + (RA+RB) (v_{out}[t] - R2 q2_{as}'[t]))}{(RA+RB) (D0_{as} + x_{as}[t])} \\
 q1_{as}'[t] + q2_{as}'[t] &== 0 \\
 K_{as} x_{as}[t] + DAMP_{as} x_{as}'[t] + E2_{as} MDYN_{as} u_{m2}''[t] &== 0 \\
 u_{m2}[0] &== 0 \\
 u_{m2}'[0] &== 0 \\
 q1_{as}[0] &== 0 \\
 q2_{as}[0] &== 0 \\
 x_{as}[0] &== 0
 \end{aligned}$$

Figure 40: Simplified DAE system.

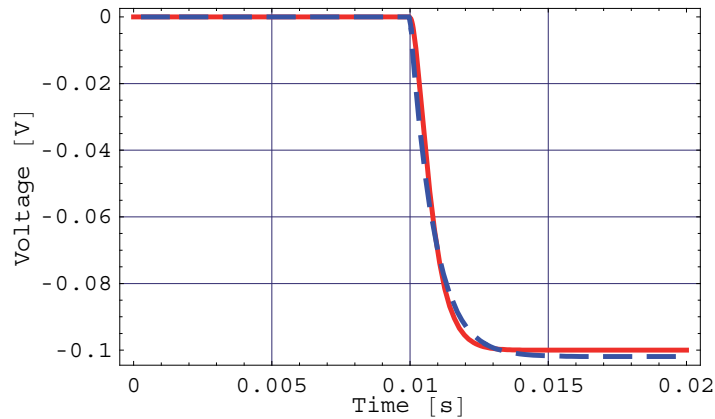


Figure 41: Comparison of original (solid red line) and simplified model (dashed blue line).

6. Approximation Example for a Mechanical System

The mechanical model used in the previous section is just a very simple model, considering the basic principles of an acceleration sensor. Now we will discuss a more detailed model for the mechanical parts, described as the appropriate composition of basic beam elements to form a network-type model. One aim is to derive a symbolic expression yielding the force constant in the static case as a function of the material properties and geometric parameters. **Fig. 42** shows a sketch of the structure for a typical micromechanical acceleration sensor (e.g. [12, 33]). The displacement u in node C3 can be considered as a function of the acceleration force F .

With these assumptions it is sufficient to consider the dark gray parts only. The other parts are important to model the full dynamic behavior, especially to calculate the eigenfrequencies. The corresponding model to analyze the behavior consists of 14 beam elements (three different kinds), two anchors and a force F acting at node C3 in y direction. The material properties (ρ , E and F) and width w are the same for all beam elements. The length and height of the two central beam elements (C3 to C4 type) are l_B and h_B whereas the height and length of the other beam elements are h_A and l_{A1} (C1 to R1 type) and l_{A2} (R1 to R2 type). This system can be described by a netlist, utilizing the beam element model described in Section 3.1.

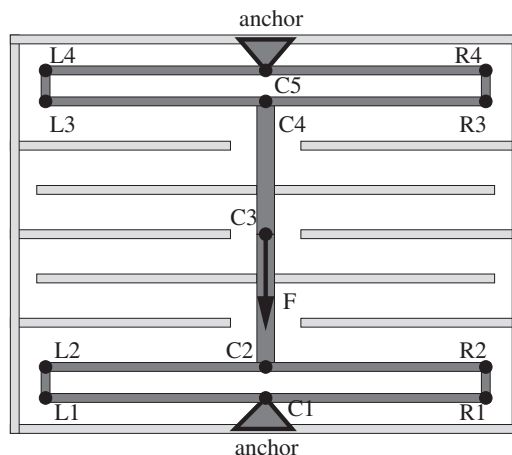


Figure 42: Micromechanical acceleration sensor.

To derive an approximated symbolic expression for the force-displacement relation in C3 the low-frequency behavior is used and further simplified to the static case. Vibrational analysis in the mechanical domain corresponds to AC analysis or small-signal analysis in the electronic domain, for which the necessary functionality is already implemented within *Analog Insydes*, so that all following steps are already automated:

- *Equation Setup:*

The full transient model of the system is created, resulting in a DAE system describing the dynamics of the 264 variables in the general case, whereof 78 variables occur in their second-order time derivative.

- *Algebraic Simplification:*

Using variable elimination and the removal of independent blocks, the system can be reduced to a DAE system in 39 variables, whereof all variables occur in their second-order time derivative.

- *Laplace Transformation:*

In our application example, the initial DAE system is linear in its variables. This results in a linear system of equations with the Laplace frequency s as an additional parameter. Analog Insydes allows to automatically deriving small-signal equations from the large-signal DAE system.

- *Define Analysis Task:*

For the original system, the transfer function $H(s)$ is computed numerically and the result is shown in **Fig. 43**. As we are mainly interested in the low-frequency behavior below 2 kHz and in the static transfer characteristic, we choose a maximum error of 10% for u at $s = 2\pi if$ where $f = 10\text{Hz}$.

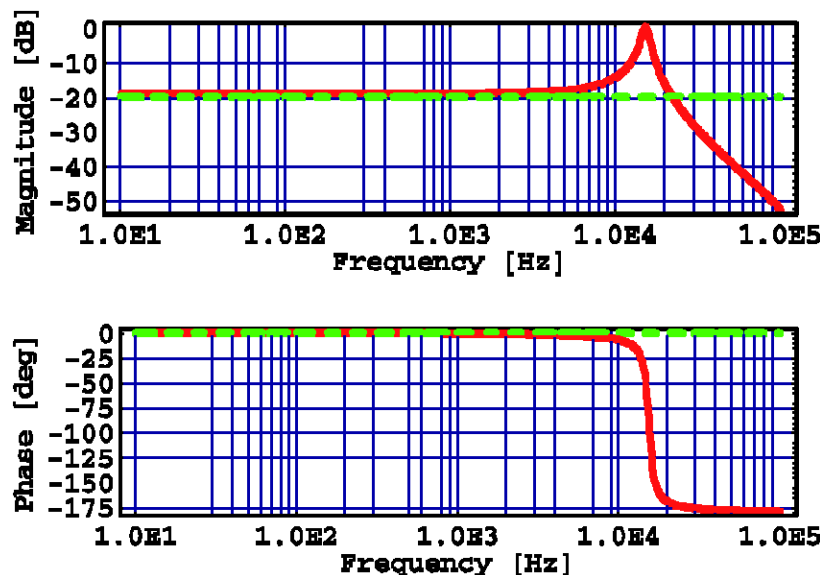


Figure 43: Comparison of original (solid red line) and simplified model (dashed green line).

- *Symbolic Approximation and Symbolic Solution:*

Insignificant terms within the symbolic matrix of the equation system are automatically identified and removed. Independent blocks of the remaining

equation system are eliminated. This leads to a system of only 10 symbolic linear equations. The numerical validation for this approximation in comparison to the original model is shown in **Fig. 44**. Of course, for the high-frequency behavior the approximation is not valid, because only the low-frequency behavior has been taken into account within the error-checking of the approximation step. This simplified system can be solved symbolically:

$$\mathbf{u} = F \frac{l_{A1}^3}{Eh_A^3 w} \frac{l_{A1} + 2l_{A2}}{2l_{A1} + l_{A2}} \quad (39)$$

- *Numerical Validation:*

Fig. 44 shows a numerical validation of the approximated expression for the static transfer characteristic in comparison to the numerical solution of the original model with respect to material and geometrical parameters. The corresponding parameters have been varied over a range of 10% to 1000% of the nominal value:

$$\begin{aligned} 22\mu\text{m} < l_{A1} < 2.2\text{mm} \\ 1.2\mu\text{m} < l_{A2} < 120\mu\text{m} \\ 13 \frac{\text{mN}}{\mu\text{m}^2} < E < 1.3 \frac{\text{N}}{\mu\text{m}^2} \\ 5\mu\text{m} < h_B < 500\mu\text{m} \end{aligned} \quad (40)$$

The figure shows the excellent correspondence between the approximated formula and the results obtained from the original model. This holds for variations of geometrical parameters like l_{A1} and l_{A2} as well as for material properties like the elastic modulus E . Additionally, parameters with minor influence on the displacement (*e.g.* h_B) do not appear within the approximate formula.

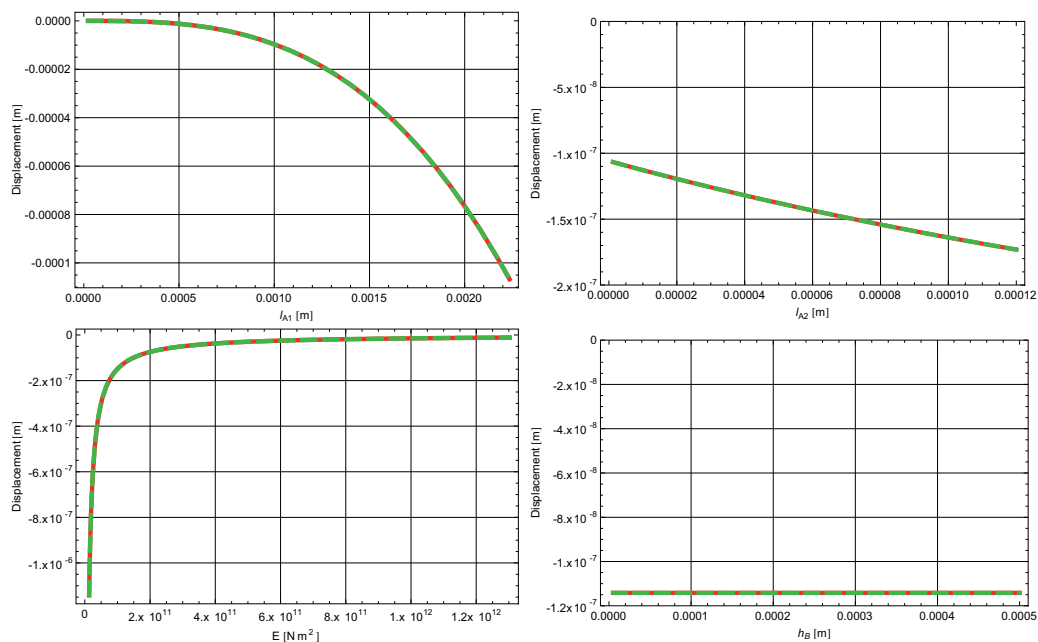


Figure 44: Validation of the approximation - original model (solid red line) and approximation (dashed green line).

7. Conclusion

In this Chapter a general modeling approach for different domains as well as multi-physical systems is presented. Its fundamentals are the generic approaches of: a branch-oriented modeling of system components; a branch-oriented topology description based on the principles of network analysis; and the application of symbolic methods. Those have already been successfully applied to electronic systems, and corresponding modeling tools are available. In order to transfer the methodology to other domains and multi-physical systems, vector-valued variables and adapted device models for basic system components have been developed and implemented. The capabilities of this new approach have been demonstrated on applications for the modeling and analysis of mixed electronic and mechanical systems.

Acknowledgements

The authors would like to thank the *Fraunhofer* society for the financial support in an internal research project. We thank especially Christoph Clauß and Roland Martin from the *Fraunhofer IIS/EAS* and Thomas Halfmann from the *Fraunhofer ITWM* for the excellent cooperation within this project.

References

- [1] K.-J. Bathe, *Finite element procedures*. Englewood Cliffs: Prentice-Hall, 1996.
- [2] C. Bakalar, "A hardware description language for analog and mixed-signal applications", *IEEE Transactions on Circuits and Systems II*, vol. 46 (1999), pp. 1263-1272.
- [3] P. Fritzon, *Principles of object-oriented modeling and simulation with Modelica 2.1*. Chichester: Wiley, 2004.
- [4] J. Broz, A. Dreyer, T. Halfmann, E. Hennig, M. Thole, T. Wichmann, *Analog insydes 2.1 Manual*. Kaiserslautern: Fraunhofer ITWM, 2005.
- [5] Fraunhofer ITWM, "Analog insydes homepage", March 2010. [Online] Available: www.analog-insydes.de [Accessed March 03, 2010].
- [6] S.-M. Chang, J.F. MacKay and G.M. Wierzbza, "Matrix reduction and numerical approximation during computation techniques for symbolic analog circuit analysis", in *IEEE International Symposium on Circuits and Systems*, San Diego, 1992, pp- 1153-1156.
- [7] D. Platte, R. Sommer, E. Barke, "An approach to analyze and improve the simulation efficiency of complex behavioral models", in *IEEE Behavioral Modeling and Simulation Conference*, San José, Sept. 2006, pp. 79-84.
- [8] D. Platte, R. Sommer, J. Broz, A. Dreyer, T. Halfmann, E. Barke, "Automatic nonlinear behavioral model generation using sequential equation structures", in *International Workshop on Symbolic Methods and Applications to Circuit Design*, Florence, 2006.
- [9] T. Wichmann, M. Thole, "Computer aided generation of analytic models for nonlinear function blocks", in *International Workshop on Power and Timing Modeling Optimization and Simulation*, Göttingen, 2000, pp. 327-335.
- [10] H. E. Koenig, W. A. Blackwell, *Electromechanical system theory*, New York: McGraw-Hill, 1961.
- [11] K. Reinschke; P. Schwarz: *Verfahren zur rechnergestützten Analyse linearer Netzwerke*. Berlin: Akademie-Verlag, 1976.
- [12] R. Neul, U. Becker, G. Lorenz, P. Schwarz, J. Haase, S. Wünsche, "A modeling approach to include mechanical microsystem components into the system simulation", in *Design, Automation and Test in Europe*, Paris, 1998, pp. 510-517.
- [13] S. Wolfram, *The mathematica book*, 5th edition, Wolfram Media Incorporated, 2003.
- [14] G. Gielen, W. Sansen, *Symbolic analysis for automated design of analog integrated circuits*. Boston: Kluwer Academic Publishers, 1991.
- [15] S. Seda, M. Degrauwe, W. Fichtner, "A symbolic analysis tool for analog circuit design automation", in *IEEE International Conference on Computer-Aided Design*, Santa Clara, 1988, pp. 488-491.

- [16] M. Kole, "Algorithms for symbolic circuit analysis based on determinant calculations", Ph.D. Thesis, University of Twente, Twente, The Netherlands, 1996.
- [17] P. Wambacq, G. Gielen, W. Sansen, "A cancellation-free algorithm for the symbolic simulation of large analog circuits", in *IEEE International Symposium on Circuits and Systems*, San Diego, 1992, pp. 1157-1160.
- [18] E. Hennig, J.-M. Tweert, R. Sommer, "Enhanced symbolic matrix approximation techniques", in *International Workshop on Symbolic Methods and Applications in Circuit Design*, Kaiserslautern, 1998, pp. 199-206.
- [19] J.J. Hsu, C. Sechen, "Fully symbolic analysis of large analog integrated circuits", in *IEEE Custom Integrated Circuit Conference*, San Diego, 1994, pp. 457-460.
- [20] J. D. Rodríguez-García, O. Guerra, E. Roca, F. V. Fernández, A. Rodríguez-Vázquez, "A new simplification before and during generation algorithm", in *International Workshop on Symbolic Methods and Applications to Circuit Design*, Kaiserslautern, 1998, pp. 110-124.
- [21] R. Sommer, E. Hennig, G. Dröge, E.-H. Horneber, "Equation-based symbolic approximation by matrix reduction with quantitative error prediction", *Alta Frequenza – Rivista di Elettronica*, Vol. 5, No. 6, 1993, pp. 29-37.
- [22] A. S. Householder, *The theory of matrices in numerical analysis*. New York: Blaisdell Publishing Co., 1964.
- [23] J. M. Ortega, W. C. Rheinboldt, *Iterative solution of nonlinear equations in several variables*, New York: Academic Press, 1970.
- [24] E. Hennig, R. Sommer, "A Reliable iterative error tracking method for approximate symbolic pole/zero analysis", in *European Conference on Circuit Theory and Design*, Vol. I, Espoo, 2001, pp. 193-196.
- [25] T. Halfmann, T. Wichmann, "Symbolic methods in industrial analog circuit design", in A.M. Anile, G. Ali, G. Mascali, Ed., *Scientific Computing in Electrical Engineering*, Capo D'Orlando: Springer 2006, pp. 87-92.
- [26] T. Wichmann, R. Popp, W. Hartong, L. Hedrich, "On the simplification of nonlinear DAE systems in analog circuit design", in *Computer Algebra in Scientific Computing*, Munich, 1999, pp. 485-498.
- [27] T. Wichmann, "Symbolische Reduktionsverfahren für nichtlineare DAE-systeme", Ph.D. Thesis, Technical University of Kaiserslautern, Kaiserslautern, Germany, 2004, Aachen: Shaker Verlag, 2004.
- [28] L. Nähnke, V. Burkhay, L. Hedrich, E. Barke, "Hierarchical automatic behavioral model generation of nonlinear analog circuits based on nonlinear symbolic techniques", in *Design, Automation and Test in Europe*, Paris, 2004, pp. 442-447.
- [29] R. März, "Numerical methods for differential-algebraic equations", *Acta Numerica*, 1992, pp. 141-198.
- [30] J. Mohring, J. Hoffmann, T. Halfmann, A. Zemitis, G. Basso, P. Lagoni, "Automated model reduction of complex gas pipeline networks", in *Pipeline Simulation Interest Group*, Palm Springs, 2004.
- [31] J. Broz, C. Clauß, T. Halfmann, P. Lang, R. Martin, P. Schwarz, "Automated symbolic model reduction for mechatronical systems", in *IEEE International Symposium on Computer-Aided Control Systems Design*, München, 2006, pp. 408-415.
- [32] S. Wüstling, O. Fromheim, H. Gemmeke, "Synergie von Elektronik und Mikromechanik am Beispiel eines Beschleunigungssensors", in *Methoden und Werkzeuge zum Entwurf von Mikrosystemen*, Frankfurt a. M., 1996, pp. 176-183.
- [33] M. Bao, *Handbook of sensors and actuators: Micromechanical transducers, pressure sensors, accelerometers and gyroscopes*. Amsterdam: Elsevier Science, 2000.

CHAPTER 13**Nonlinear Template-Free Symbolic Performance Modeling for Design and Process Variation Analysis of Analog Circuits****Trent McConaghy^{1,*} and Georges G.E. Gielen²**

¹*Solido Design Automation Inc., Canada (formerly with Katholieke Universiteit Leuven, Belgium) and*
²*Katholieke Universiteit Leuven, Belgium*

Abstract: This chapter presents the CAFFEINE tool, which is a method for generating symbolic performance models of electronic circuits without any prior specification of an equation template. CAFFEINE uses SPICE simulation data, allowing it to handle strongly nonlinear circuits, statistical process variations, and a variety of analysis types. CAFFEINE expressions are canonical form functions: product-of-sum layers alternate with sum-of-product layers, as defined by a context-free grammar. Besides the attribute of interpretability, CAFFEINE models demonstrate lower prediction error than several state-of-the-art regression techniques including posynomials, projection-based quadratic models, boosted neural networks, piecewise polynomials / splines, kriging, and support vector machines. In addition, CAFFEINE is also useful in variation-aware modeling, behavioral modeling, and tradeoff modeling.

Keywords: Symbolic analysis, symbolic modeling, template-free modeling, performance modeling, CAFFEINE, canonical form function, SPICE accuracy, interpretability, integrated circuits, grammar, genetic programming, evolutionary algorithm, posynomial, regression, whitebox modeling, blackbox modeling, response surface modeling, template extraction.

1. Introduction

Both *symbolic analysis* and *symbolic modeling* aim to derive human-interpretable expressions of analog circuit behavior [1]. Symbolic analysis extracts the expressions *via* topological analysis of the circuit, whereas symbolic modeling extracts the expressions by using SPICE simulation data. These expressions have the same applications: knowledge acquisition and educational / training purposes, analytic model generation for automated circuit sizing, design space exploration, repetitive formula evaluation including statistical analysis, analog fault diagnosis and testability analysis, and analog behavioral model generation [2]. In particular, a tool that can help a designer improve his understanding of a circuit is highly valuable, because it leads to better decision-making in circuit sizing, layout, verification, and topology design, regardless of the degree of automation. Therefore, approaches to generate symbolic expressions are of great interest.

Historically, symbolic analysis came first, starting with ISAAC [3] and followed by several other techniques; see chapter 1 in this book for a review. Until recently, the main weakness was their limitation to linear and weakly nonlinear circuits. This was overcome *via* piecewise-linear/polynomial modeling approaches (*e.g.* [4, 5]), but at the cost of interpretability.

Leveraging SPICE simulations in modeling is promising because simulators readily handle nonlinear circuits, environmental effects (*e.g.* temperature, power supply voltage, loads), manufacturing effects, different technologies, new effects (*e.g.*

*Address correspondence to Trent McConaghy: Solido Design Automation Inc., Canada (formerly with Katholieke Universiteit Leuven, Belgium); E-mail: gtrenc@gmail.com

proximity, electromigration, packaging), and more. From simulation data, a model $y=f(\mathbf{x})$ is constructed, where y is typically a performance metric, \mathbf{x} includes design, process, or environmental variables, and f is an approximation of the SPICE mapping. Models used include linear models, posynomials, polynomials, splines, neural networks, support vector machines, Latent Variable Regression (LVR), and kriging (see refs. in [6]). However, such models either follow an overly restrictive functional template which limits their applicability, or they are opaque and thus provide no insight to the designer. Less opaque flows exist, such as visualizing CART trees [7]; nonlinear sensitivity analysis [7]; or plotting the mapping from LVR affine transform $\mathbf{w}_1 * \mathbf{x}$ to the output $y=f_l(\mathbf{w}_1 * \mathbf{x})$ [8]. While useful, these approaches do not give the functional relations that symbolic models provide.

The aim of *symbolic modeling* as defined in this chapter is to use simulation data to generate *interpretable mathematical expressions* for circuit applications, typically relating the circuit performances to the design variables. Symbolic modeling has similar goals to symbolic analysis, but a different core approach to solving the problem. In [9] posynomial-based symbolic models are constructed. The main problem is that the models are constrained to a predefined template, which restricts the functional form. Also, the models have dozens of terms, limiting their interpretability for designers. Finally, the approach assumes posynomials can fit the data; in analog circuits there is no guarantee of this. There have also been advances in quadratic modeling [10], but polynomials also have a restrictive structure.

This chapter describes CAFFEINE, which generates symbolic models having more *open-ended* functional forms (*i.e.* without a pre-defined template), for arbitrary nonlinear circuits and circuit characteristics, and at the same time ensuring that the models are *interpretable*. **Fig. 1** shows a target flow that reflects these goals. CAFFEINE treats the task as a search problem in the space of possible functional form trees. An appropriate search algorithm is then Genetic Programming (GP) [11]. Within GP search, a *grammar* [12] constrains the generated functions to those that are human-interpretable. CAFFEINE stands for Canonical Functional Form Expressions in Evolution [6, 13].

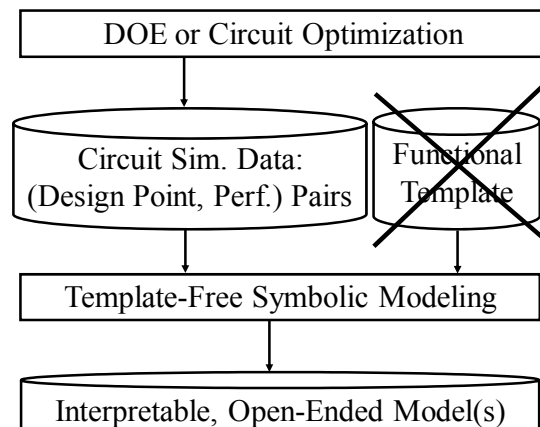


Figure 1: Template-free symbolic modeling flow.

Some features of CAFFEINE include:

- To our knowledge, it is the only tool for *template-free* symbolic modeling;

- Because it uses SPICE data, it can model any nonlinear circuit, under any analysis (*e.g.* ac, dc, transient, noise), any performance characteristic (*e.g.* power consumption, slew rate, even yield and Cpk), and inputs controllable or uncontrollable variables (*e.g.* design variables, process variables, environmental variables, parasitic variables);
- It returns models that are compact and understandable, yet with good accuracy. In fact, it returns a *set* of possible models that *trade off* accuracy and complexity by using multi-objective search [14].

This chapter is organized as follows. Section 2 presents the problem formulation. Section 3 presents background on genetic programming. Section 4 introduces the heart of CAFFEINE: canonical form functions. Section 5 describes the reference search algorithm, which uses multi-objective genetic programming and a grammar to constrain to canonical form functions. Section 6 gives some experimental results. Section 7 describes other applications of CAFFEINE. Section 8 concludes.

2. Problem Formulation

The symbolic performance modeling problem that we address has the flow of **Fig. 1**. Its inputs and outputs are as follows:

Given:

- X and y : A set of $\{\mathbf{x}_j, y_j\}$, $j=1..N$ data samples where \mathbf{x}_j is a N_d -dimensional design point j and y_j is a corresponding circuit performance value measured from SPICE simulation of that design. Design of Experiments (DOE) [15] or circuit optimization can be used to generate the data samples.
- No model template.

Determine:

- A set of symbolic models M that provide the Pareto-optimal tradeoff between minimizing model complexity f_1 and minimizing future model prediction error f_2 .

The formulation used is a constrained optimization problem:

$$M = \min \left\{ \begin{array}{l} f_1 = \text{complexity}(m) \\ f_2 = E_{x,y}L(y, m(x)) \end{array} \right\} \quad \text{s.t. } m \in \Psi \quad (1)$$

where Ψ is the space of template-free symbolic models. The algorithm will traverse Ψ to return a Pareto-optimal set $M = \{m_1, m_2, \dots, m_{N_M}\}$. Each model m maps an N_d -dimensional input \mathbf{x} to a scalar circuit performance approximation \hat{y} , *i.e.* $\hat{y} = m(\mathbf{x})$. Complexity is *some* measure that differentiates the degrees of freedom between different models (see (5)). $E_{x,y}L$ is the expected loss for a given m over future predictions in the distribution pdf(\mathbf{x}), where L is the squared-error loss function [16]:

$$L(y, m(x)) = (y - m(x))^2 / 2 \quad (2)$$

Section 5.1 describes how an approximation for $L()$ is computed. By definition, no model in the Pareto-optimal set M dominates any other model. A model m_a “dominates” another model m_b if $\{f_j(m_a) \leq f_j(m_b)\} \forall j$, and $\{f_j(m_a) < f_j(m_b)\} \exists j$. In our case, $j = \{1, 2\}$. That is, to be Pareto-optimal, a model must be at least as good as any other model on both objectives, and better than any model in one objective.

3. Background: Genetic Programming

Genetic Programming (GP) [11] is an evolutionary algorithm, with the distinguishing characteristic that GP individuals (points in the design space) are *trees*. Since a symbolic model is a function and can be represented as a tree, the search for the above models can be accomplished by GP search.

The functional form of results from canonical GP is completely unrestricted. While this sounds promising compared to the restrictions of fixed-template regression, it actually goes a little too far: an unrestricted form is almost always difficult to analyze. GP-evolved functions can be notoriously *complex* and *un-interpretable*. For example, [11] showed functions so bloated [17] that they take up a full page of dense text. A recent paper complains: “[GP-evolved] expressions can get, as we have seen, quite complex, and it is often extremely difficult to understand them without a fair bit of interaction with a tool such as *Mathematica*” [18].

We can see for ourselves. Using a dataset from section 6, canonical GP evolution returned the following expression¹:

```
- 1.40 * ( vsg1 + max( vsg5, max( max( max( vsg5, max(
vsg3 + vgs2, min( vsg3, abs( 1/vds2 ) ) ) ) - log10(vsd5)
), min( ib2, abs( sqrt( abs(id1) ) ) ) ) - log10(vsd5),
max( id2, min( vsg3, abs( sqrt( abs( log10(id2) ) ) ) ) )
+ log10(vsd5) ) - min( vsg3, abs( sqrt( abs(id1) ) ) ) -
log10(vsd5) ) )
```

Improvements in interpretability are clearly needed. The next section presents CAFFEINE to handle this issue.

4. CAFFEINE Canonical form Functions

The design of CAFFEINE follows two guidelines: 1) ensure maximum expressiveness per node, and 2) make all candidate functions directly interpretable.

Fig. 2 shows the general structure of a CAFFEINE function. It alternates between levels of *sum-of-product* expressions and *product-of-sum* expressions. Each sum-of-product expression is a weighted linear add of an overall offset term plus weighted basis functions. A basis function is a combination of product terms, where each product term is a polynomial/rational, zero or more nonlinear operators, and zero or more unity operators. Each product term acts as a “gate” to the next sum-of-products layer.

Fig. 3 left shows an example function and its corresponding tree. In the “ $7.1/x_3$ ” part of the function, the 7.1 is the tree's top left “ w_0 ” and the “ $1/x_3$ ” is its neighboring “poly/rat'l of vars”. The “1.8” corresponds to top “ w_1 ”, and the “ x_1 ” is its neighboring “poly/rat'l of vars”. The function's “log” corresponds to “nonlinear func”, which in the tree holds the “weighted linear add” term “ $-1.9 + 8.0/x_1 + 1.4 * x_2^2 / x_3$ ”. That term itself breaks down: function's the “-1.9” is the tree's lower “ w_{offset} ”; “ $8.0/x_1$ ” corresponds to the tree's lower left “ w_0 ” * “poly/rat'l of vars”; and “ $1.4 * x_2^2 / x_3$ ” corresponds to the tree's lower right “ w_1 ” * “poly/rat'l of

¹ The expression font and style are presented like [11].

vars". Note how CAFFEINE places coefficients only where they are needed, and nowhere else.

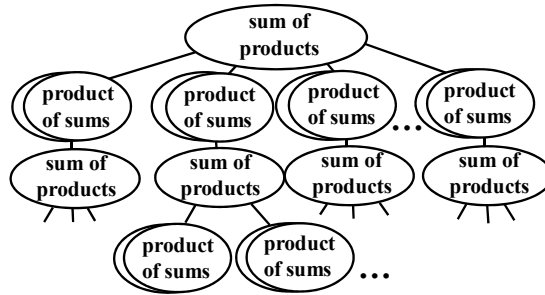


Figure 2: CAFFEINE evolves functions of this canonical form. While it can go deeper indefinitely, it is typically only as deep as shown in order to retain human interpretability.

Fig. 3 right gives an example which has unity functions for product terms. Note how there is *no* nonlinear function that gates one layer of linear adds to the next -- this is how CAFFEINE supports a product-of-sums formulation.

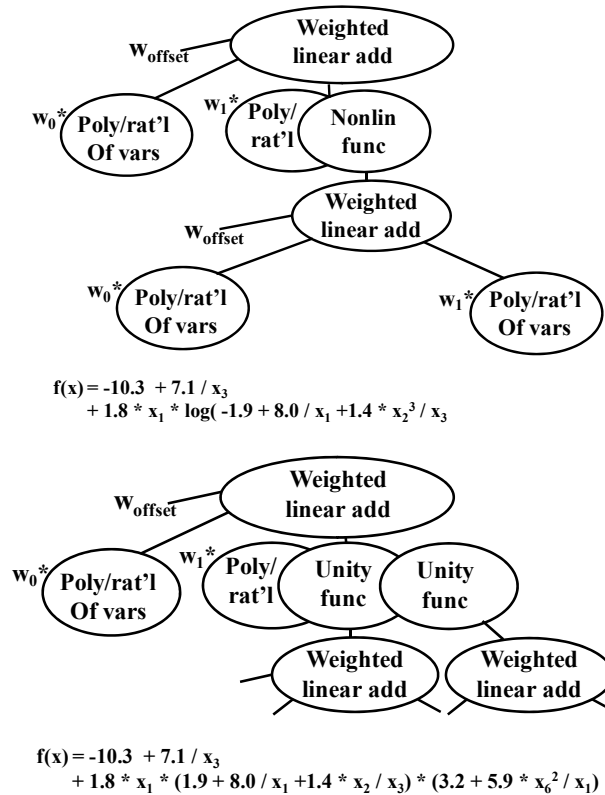


Figure 3: Top: Example of a function in text form, and its corresponding CAFFEINE tree form. Bottom: Example where CAFFEINE product terms include unity functions.

Typical usage of CAFFEINE would restrict the number of product term layers to just one or two (like in **Fig. 2**), therefore ensuring that there is not an excessive compounding of nonlinear components such as $\log(\sin(\exp(x)))$. There can also be a limit on the maximum number of basis functions. Due to the use of a canonical form, all evolved functions are immediately interpretable, with no symbolic manipulation needed.

5. CAFFEINE Search Algorithm

This section describes the search algorithm used on CAFFEINE functions. CAFFEINE search uses multi-objective grammatically-constrained Genetic Programming (GP). The multi-objective aspect means it returns a set of models which trade off between error and complexity. The grammatically-constrained aspect means it follows the canonical functional forms described in the previous section.

In CAFFEINE, the overall expression is a linear function of N_B basis functions B_i ; $i=\{1,2,\dots, N_B\}$:

$$\hat{y} = m(x) = a_0 + \sum_{i=1}^{N_B} a_i * B_i(x) \quad (3)$$

A CAFFEINE individual m has one GP tree to define each basis function: $m = \{B_1, B_2, \dots, B_{N_B}\}$. The linear coefficients $a \in \mathfrak{R}^{N_B+1}$ are determined on-the-fly using linear regression on the least-squares cost function (2).

5.1. Multi-objective approach

CAFFEINE can use any competent multi-objective evolutionary algorithm; in our experiments we use NSGA-II [14]. NSGA-II returns a set of individuals that, collectively, trade off model error and complexity. Error and complexity are the objectives f_1 and f_2 in (1). The error (expected loss $E_{x,y}L$) is approximated by the “training error” ε_{tr} , which is the normalized root mean squared error of individual m on the training data:

$$\varepsilon_{tr} = \sqrt{\frac{1}{N_{tr}} * \sum_{i=1}^{N_{tr}} \left(\frac{\hat{y}_{tr,i} - y_{tr,i}}{\max(y) - \min(y)} \right)^2} \quad (4)$$

where N_{tr} is the number of training samples; $y_{tr,i}$ is sample i of training outputs \mathbf{y}_{tr} , $\hat{y}_{tr,i} = F(\mathbf{x}_{tr,i}; m)$ using (3); and $\mathbf{x}_{tr,i}$ is sample i of training inputs \mathbf{X}_{tr} . Note that the y -values are scaled by \mathbf{y} , not \mathbf{y}_{tr} . ε_{test} has a similar formula, except the N_{tr} training points $\{\mathbf{y}_{tr}, \mathbf{X}_{tr}\}$ are replaced by the N_{test} testing points $\{\mathbf{y}_{test}, \mathbf{X}_{test}\}$.

The model complexity is measured from the number of basis functions, the number of nodes in each tree, and the exponents of “variable combos” (VCs or vc), according to:

$$complexity(m) = \sum_{j=1}^{N_B} \left(w_b + nnodes(j) + \sum_{k=1}^{nvc(j)} vccst(vc_{k,j}) \right) \quad (5)$$

where w_b is a constant to give a minimum cost to each basis function, $nnodes(j)$ is the number of tree nodes of basis function j , and $nvc(j)$ is the number of VCs of basis function j , with cost:

$$vccst(vc) = w_{vc} * \sum_{i=1}^{N_d} |vc(i)| \quad (6)$$

The approach accomplishes *simplification during generation* [19] by maintaining evolutionary pressure towards lower complexity. The user avoids an *a priori* decision on error or complexity because the algorithm generates a *set* of models that provide tradeoffs of alternatives, rather than producing just *one* model.

Note that specific parameter settings for the algorithm are given in the experiments (section 6).

5.2. Grammar implementation of canonical form functions

In GP, a means of constraining search is *via* a grammar, as in [12]. Tree-based evolutionary operators such as crossover and mutation must respect the derivation rules of the grammar. The CAFFEINE grammar, shown in **Table 1**, is explicitly designed to create separate layers of linear and nonlinear functions and to place coefficients and variables carefully, in adherence with **Fig. 2**.

Table 1: CAFFEINE Grammar.

REPVC	\rightarrow	VC REPVC * REPOP REPOP
REPOP	\rightarrow	REPOP * REPOP OP_1ARG (W + REPADD) OP_2ARG (2ARGS) ... 3OP, 4OP etc
2ARGS	\rightarrow	W + REPADD , MAYBEW MAYBEW , W + REPADD
MAYBEW	\rightarrow	W W + REPADD
REPADD	\rightarrow	W * REPVC REPADD + REPADD
OP_2ARG	\rightarrow	DIVIDE POW MAX ...
OP_1ARG	\rightarrow	INV LOG10 ...

First, we describe the notation of **Table 1**. The nonterminal symbols are in bold-case; terminal symbols are not. Each line (or two) shows the possible expressions that a nonterminal symbol on the left can map (\rightarrow) into. The possible expressions, *i.e.* “derivation rules” are separated by the OR operator “|”.

We now explain how the grammar implements canonical form functions. **REP** is short for “repeating”, such as “repeating operators” **REPOP** and “repeating variable combo” **REPVC**, which are explained further. The start symbol is **REPVC**, which expands into one basis function (remember that an individual has several root-level basis functions). Note the strong distinction among operators. The root is a product of variables (**REPVC**) and / or nonlinear functions (**REPOP**). Within each nonlinear function is **REPADD**, the weighted sum of next-level basis functions.

A **VC** is a “variable combo”, intended to maintain a compact representation of polynomials / rationals. Its expansion could have been implemented directly within the grammar; though in our baseline system we store a vector holding an integer value per design variable as the variable's exponent. An example vector is [1,0,-2,1], which means $x_1 * x_4 / x_3^2$, and according to (6) has cost $|1| + |0| + |-2| + |1| = 4$. This approach guarantees compactness and allows for special operators on the vector.

In determining coefficient values, we distinguish between linear and nonlinear coefficients. As described, a CAFFEINE individual is a set of basis functions which are linearly added. Each basis function is a tree of grammatical derivations. Linear coefficients are found by evaluating each tree across all input samples to get a matrix of basis function outputs, then to apply least-squares regression with that matrix and the target output vector to find the optimal linear weights.

With each nonlinear coefficient **W** in the tree (*i.e.* ones that are not found *via* linear regression), a real value will accompany it, taking a value in the range $[-2 * B, +2 * B]$. During interpretation of the tree the value is transformed into $[-1e + B, -1 - eB] \cup [0.0] \cup [1e - B, 1e + B]$. B is user-set; see section 6.1.

POW(a, b) is a^b . When the symbol 2ARGS expands to include MAYBEW, either the base or the exponent (but not both) can be constants.

5.3. High-level CAFFEINE algorithm

Table 2 gives the algorithm (*ExtractSymbolicCaffeineModels*). It takes in the training inputs X and training outputs y . It will output a Pareto-optimal set of models, M . Line 1 initializes M , the current set of parents P , and the current set of children Q , all to empty sets \emptyset . Lines 2-3 loop across the population size N_{pop} to randomly draw each individual P_i from the space of possible canonical form functions Ψ . Line 4 begins the evolutionary algorithm's (EA's) generational loop of lines 5 and 6. The loop stops when the target number of generations, $N_{gen,max}$, is hit. Line 5 does the main EA work, which here is a single generation of the NSGA-II multi-objective EA (see [14] for details). Line 6 updates the external archive of Pareto-optimal individuals, M , by nondominated-filtering on the existing M with the recently updated parents P and children Q . Line 7 concludes the routine, by returning the Pareto-optimal symbolic models, M .

Table 2: Procedure *ExtractSymbolicCaffeineModels*.

Inputs: X, y
Outputs: M
1. $M = \emptyset; P = \emptyset; Q = \emptyset$
2. for $i = 1..N_{pop}$:
3. $P_i =$ random draw from Ψ
4. for $N_{gen} = 1..N_{gen,max}$:
5. $\{P, Q\} = OneNsgaiiGeneration(P, Q)$
6. $M = nondominatedFilter(M \cup P \cup Q)$
7. return M

5.4. Evolutionary search operators

We now describe how trees are randomly generated, and explain the search operators on the trees. The search operators are grouped by the aspect of search representation that they concern: grammar, real-valued coefficient, Variable Combos (VCs), and basis functions.

Random generation of trees and subtrees from a given symbol involves merely randomly picking one of the derivations of one of the symbols, and recursing the (sub)tree until terminal symbols are encountered (subject to tree depth limits).

Grammatical restrictions on the trees lead to a natural grammar-obeying crossover operator and mutation operator, as described by Whigham [12]. Whigham-style crossover works as follows: it randomly picks a node on the first parent, then randomly picks a node on the second parent with the constraint that it must be the same grammatical symbol (*e.g.* REPOP) as the first node, and finally swaps the subtrees corresponding to each node. Whigham-style mutation involves randomly picking a node, then replacing its subtree with a randomly-generated subtree (as in the generation of initial trees).

Real-valued coefficients are mutated according to a Cauchy distribution [20], which cleanly combines aggressive local tuning with the occasional large change.

VCS have the operators: one-point crossover, and randomly adding or subtracting to an exponent value.

Each individual has a list of basis functions, which also leads to special operators: creating a new individual by randomly choosing >0 basis functions from each of 2 parents; deleting a random basis function; adding a randomly generated tree as a basis function; copying a subtree from one individual to make a new basis function for another.

6. Experimental Results

This section describes the application of CAFFEINE to building symbolic models for analog circuits that map design variables to performances, for problems with up to 13 input variables. It shows the actual symbolic models generated, measured error *versus* complexity tradeoffs, how prediction error and complexity compare to posynomials, and how prediction error compares to other state-of-the-art (blackbox) regression approaches.

6.1. Experimental setup

Unary operators allowed are: $\text{sqrt}(x)$, $\log_{10}(x)$, $1/x$, x^2 , $\sin(x)$, $\cos(x)$, $\tan(x)$, $\max(0,x)$, $\min(0,x)$, 2^x , and 10^x , where x is an expression. Binary operators allowed are $x_1 + x_2$, $x_1 * x_2$, $\max(x_1, x_2)$, $\min(x_1, x_2)$, $x_1 \wedge x_2$, and x_1/x_2 . Conditional operators include $\leq(\text{testExpr}, \text{condExpr}, \text{exprIfLessThanCond}, \text{elseExpr})$ and $\leq(\text{testExpr}, 0, \text{exprIfLessThanCond}, \text{elseExpr})$. Any input variable could have an exponent in the range $\{\dots -2, -1, 1, 2, \dots\}$. While real-valued exponents could have been used, that would have harmed interpretability.

The circuit being modeled in this example is a high-speed CMOS Operational Transconductance Amplifier (OTA) as shown in **Fig. 4**. The goal is to discover expressions for the following 6 performance characteristics: the low-frequency gain (A_{LF}), unity-gain frequency (FU), Phase Margin (PM), input-referred offset voltage (V_{offset}), and the positive and negative slew rate (SR_p , SR_n). To allow a direct comparison to the posynomial approach [9], an almost-identical problem setup was used, as well as identical simulation data. The only difference is that, because scaling makes the model less interpretable, neither the inputs nor the outputs were scaled. The one exception is that FU is log-scaled so that the mean-squared error calculations and linear learning are not wrongly biased towards high-magnitude samples of FU . The technology is $0.7 \mu\text{m}$ CMOS. The supply voltage is 5V. $V_{th,nom}$ is 0.76V and -0.75V for the NMOS and PMOS devices, respectively. The load capacitance is 10 pF.

Good training data is essential to the methodology. The choice of design variables and sampling methodology determines the extent to which the designer can make inferences about the physical basis, and what regions of design space the model is valid in. We used an operating-point driven formulation [21], where currents and transistor gate drive voltages comprise design variables (13 variables in our case). Device sizings could have been used as design variables instead; it all depends on designer preference. Full orthogonal-hypercube Design-Of-Experiments (DOE) [15] sampling of design points was used, with scaled $dx=0.1$ (where dx is % change in variable value from the center value) to have 243 samples. The simulation time for one sample was about 1 s, or 4 min for all samples; this is fully dependent on the circuit, analyses, and experimental design method being used. These samples, otherwise unfiltered, were used as training data inputs. Testing data inputs were also

sampled with full orthogonal-hypercube DOE and 243 samples, but with $dx = 0.03$. Thus, in this experiment we are creating a somewhat localized model; one could just as readily model a broader design space, but this allows us to compare the results to [9].

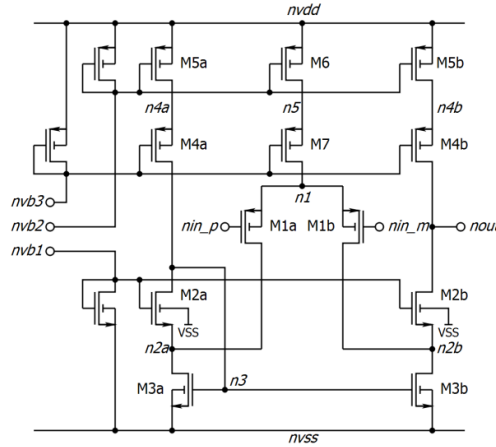


Figure 4: CMOS high-speed OTA used to illustrate the performance modeling in CAFFEINE.

The run settings were: N_B = maximum number of basis functions = 15 (any larger is definitely non-interpretable), N_{pop} = population size = 200 (like NSGA-II's default), $N_{gen,max}$ = 5000 generations (more than enough time to converge), maximum tree depth = 8 (so that each basis function has exactly one layer of nonlinear operators), and \bar{w} coefficients range $[-1e+10, -1e-10] \cup [0.0] \cup [1e-10, 1e+10]$ (i.e. $B=10$; therefore coefficients can cover 20 orders of magnitude, both positive and negative). All operators had equal probability (a reliable setting), except parameter mutation was 5x more likely (to encourage tuning of a compact function). The complexity measure settings were $w_b = 10$, $w_{vc} = 0.25$. That is, the cost of adding a basis function is relatively high compared to the cost of adding another variable combo.

One run was done for each performance goal, for 6 runs total. The Python-based implementation took approximately 10 minutes per run. We would expect a C-based implementation to be up to 10x faster yet.

We calculate the normalized mean-squared error on the training data ε_{tr} and on the separate testing data ε_{test} as described in (4). These are standard measurements of model quality in regression literature. The testing error ε_{test} is ultimately the more important measure, because it measures the model's ability to generalize to unseen data. These measures are identical to two of the three posynomial “quality of fit” measures in [9]: its measure “worst-case quality” q_{wc} is the training error ε_{tr} , and its measure “typical case quality” q_{tc} is ε_{test} (as long as the constant c in the denominator is set to zero, which [9] did.)

6.2. Results: whitebox models and tradeoffs

Let us first examine some symbolic models generated by CAFFEINE. We ask: “which symbolic models have <10% training and testing error, with the lowest complexity?” **Table 3** shows those functions. (Note that FU has been converted to its true form by putting the generated function to the power of 10). We see that each

function has up to four basis functions, not including the constant. For V_{offset} a constant was sufficient to keep the error within 10%. We see that a rational functional form was favored heavily; at these target errors only one nonlinear function, $\ln()$, appears (for A_{LF}). The $\ln()$ indicates that the order of magnitude of some input variables is meaningful.

Table 3: CAFFEINE-Generated Symbolic Models which Have <10% Training and Testing Error.

Perf.	Expression
A_{LF}	$-10.3 + 7.08e-5 / id1$ $+ 1.87 * \ln(-1.95e+9 + 1.00e+10 / (vsg1*vsg3))$ $+ 1.42e+9 *(vds2*vds5) / (vsg1*vgs2*vsg5*id2)$
f_u	$10^{(5.68 - 0.03 * vsg1 / vds2 - 55.43 * id1 + 5.63e-6 / id1)}$
PM	$90.5 + 190.6 * id1 / vsg1 + 22.2 * id2 / vds2$
V_{offset}	$- 2.00e-3$
SR_p	$2.36e+7 + 1.95e+4 * id2 / id1 - 104.69 / id2 + 2.15e+9 * id2 + 4.63e+8 * id1$
SR_n	$- 5.72e+7 - 2.50e+11 * (id1*id2) / vgs2 + 5.53e+6 * vds2 / vgs2 + 109.72 / id1$

One can examine the equations in more detail to gain an understanding of how design variables in the topology affect performance. For example, A_{LF} is inversely proportional to i_{dl} , the current at the OTA's differential pair. Or, SR_p is solely dependent on i_{d1} and i_{d2} and the ratio i_{d1}/i_{d2} . Or, within the design region sampled, the nonlinear coupling among the design variables is quite weak, typically only as ratios for variables of the same transistor. Or, that each expression only contains a (sometimes small) subset of design variables. Or, that transistor pairs M1 and M2 are the only devices affecting five of the six performances (within 10% error).

We now examine the CAFFEINE-generated tradeoffs between training error ε_{tr} (q_{wc}) and complexity, which **Fig. 5** illustrates. All models in the tradeoff of training error vs. complexity are shown: as complexity increases, the training error decreases. For each performance instance, CAFFEINE generates a tradeoff of about 50 different models. As expected, a zero-complexity model (*i.e.* a constant) has the highest training error of 10-25%. The highest-complexity models have the lowest training error, of 1-3%.

We can also examine the curves relating complexity to the number of basis functions. Recall that complexity is a function of both the number of basis functions, and the complexity of each tree within each basis function. In the curves, we see that the number of basis functions usually increases with the complexity. However, sometimes the complexity increases by having larger trees within existing basis functions, rather than adding more basis functions. This can be seen in the curves: as complexity increases, the number of bases temporarily levels off, or even decreases.

The testing error, ε_{test} , is also shown as q_{tc} in **Fig. 5**. We see that unlike the training error, it is not monotonically decreasing as complexity rises. This means that some less complex models are more predictive than more complex ones. However, we can prune the models down to the ones that give a tradeoff between testing error and complexity, as shown in **Fig. 6**. These are the most interesting and useful.

It is notable that the testing error is lower than the training error in almost all cases. This sounds promising, but such behavior is rare in the regression literature, and

made us question what was happening. It turns out that there is a valid reason: recall that the training data is from extreme points of the sampling hypercube (scaled $dx=0.10$), and the testing data is internal to the hypercube ($dx=0.03$). This testing data tests the *interpolation* ability. Thus, models that really *are* predictive should be able to interpolate well, even at the cost of a perfect fit to the extreme points. In any case, validly having the testing error lower than the training error demonstrates the strength of the CAFFEINE approach.

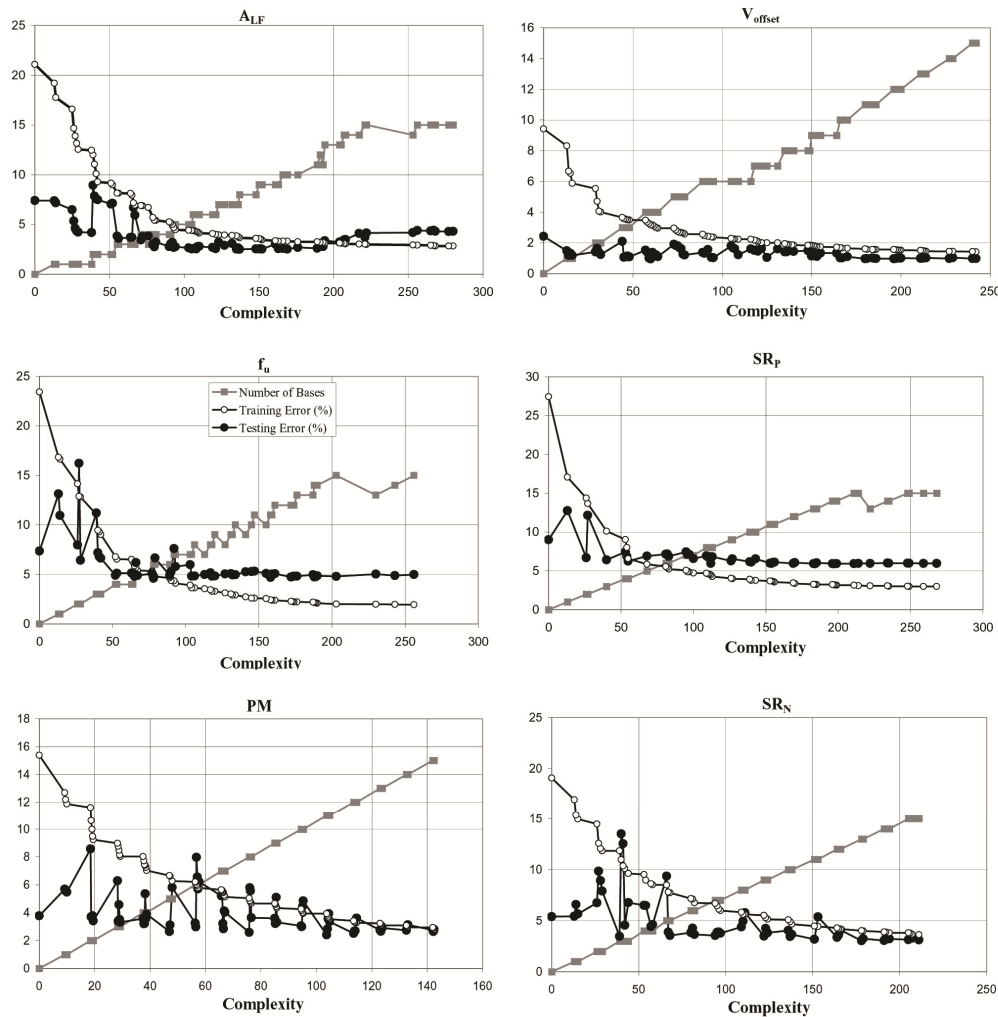


Figure 5: Plots of models' training error, testing error, and number of bases vs. the complexity for each performance goal for the OTA of **Figure 4**. Every (diamond, triangle, square) triplet corresponds to a symbolic model at a given complexity.

One may improve the understanding of the basic dependencies in a circuit in another fashion: by examining expressions of varying complexity for a single performance characteristic. Low-complexity models will show the macro-effects; alterations to get improved error show how the model is refined to handle second-order effects. **Table 4** shows models generated for the Phase Margin (PM) for decreasing training and testing error. A constant of 90.2, while giving 15% training error, had only 4% test error. For better prediction, CAFFEINE injected two more basis functions; one basis

being the current into the differential pair i_{d1} , the other basis, i_{d2}/v_{ds2} , being the ratio of the current to the drain-source voltage of M2; *i.e.* M2's small-signal output conductance ($1/r_{out2}$). The next model turns the input current term into a ratio i_{d1}/v_{gs1} ; *i.e.* M1's transconductance, inverted ($1/g_{m1}$). Interestingly, and reassuringly, almost all ratios use the same transistor in the numerator and denominator.

Table 4: CAFFEINE-Generated Models of the Phase Margin (PM) of the OTA of Fig. 4, in Order of Decreasing Error and Increasing Complexity.

Test error (%)	Train error (%)	PM Expression
3.98	15.4	90.2
3.71	10.6	$90.5 + 186.6 * id1 + 22.1 * id2 / vds2$
3.68	10	$90.5 + 190.6 * id1 / vsg1 + 22.2 * id2 / vds2$
3.39	8.8	$90.1 + 156.85 * id1 / vsg1 - 2.06e-03 * id2 / id1 + 0.04 * vgs2 / vds2$
3.31	8	$91.1 - 2.05e-3 * id2 / id1 + 145.8 * id1 + 0.04 * vgs2 / vds2 - 1.14 / vsg1$
3.2	7.7	$90.7 - 2.13e-3 * id2 / id1 + 144.2 * id1 + 0.04 * vgs2 / vds2 - 1.00 / (vsg1*vsg3)$
2.65	6.7	$90.8 - 2.08e-3 * id2 / id1 + 136.2 * id1 + 0.04 * vgs2 / vds2 - 1.14 / vsg1 + 0.04 * vsg3 / vsd5$
2.41	3.9	$91.1 - 5.91e-4 * (vsg1*id2) / id1 + 119.79 * id1 + 0.03 * vgs2 / vds2 - 0.78 / vsg1 + 0.03 * vsg1 / vsd5 - 2.72e-7 / (vds2*vds5*id1) + 7.11 * (vgs2*vsg4*id2) - 0.37 / vsg5 - 0.58 / vsg3 - 3.75e-6 / id2 - 5.52e-6 / id1$

Such analyses demonstrate the core aim of CAFFEINE symbolic modeling: gaining insight into the design-performance relationship of the circuit under analysis.

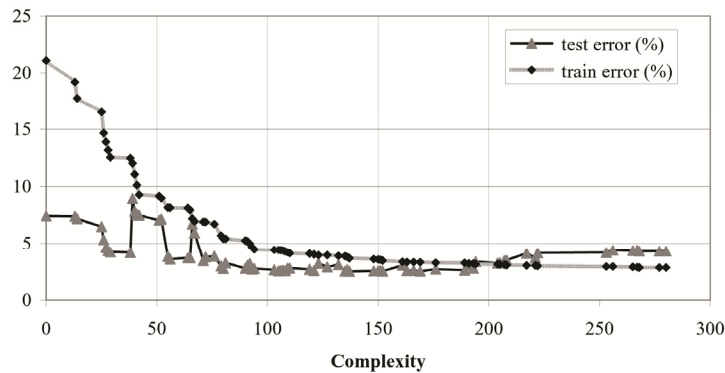


Figure 6: Every (diamond, triangle) tuple is a symbolic model for the low-frequency gain ALF like Figure 5, except filtered to only keep models on the tradeoff of testing error vs. complexity.

6.3. Results: comparison to state-of-the-art blackbox regression approaches

While other modeling techniques may produce models that are opaque (and therefore not interpretable), it is still instructive to see how well CAFFEINE compares to them in terms of prediction ability. So, on the 6 problems already described in section 6.1, we tested the following regression techniques: a constant, linear models with least-squares fit, full quadratic models with least-squares fit, projection-based quadratic models (PROBE) [10], posynomial models [9], feedforward neural networks (FFNN) [22], boosting the FFNNs, multivariate adaptive regression splines (MARS) (*i.e.* piecewise polynomial with stepwise construction) [23], least-squares support vector machines (LS-SVM) [24], and kriging [25].

Fig. 7 shows the resulting test errors for the 6 performances (originally from [27]).

On this data set CAFFEINE has the lowest average prediction error. MARS and kriging have the next-best performance. The FFNN, boosted FFNN, SVM, and linear model perform similarly. The quadratic and posynomial approaches perform the worst.

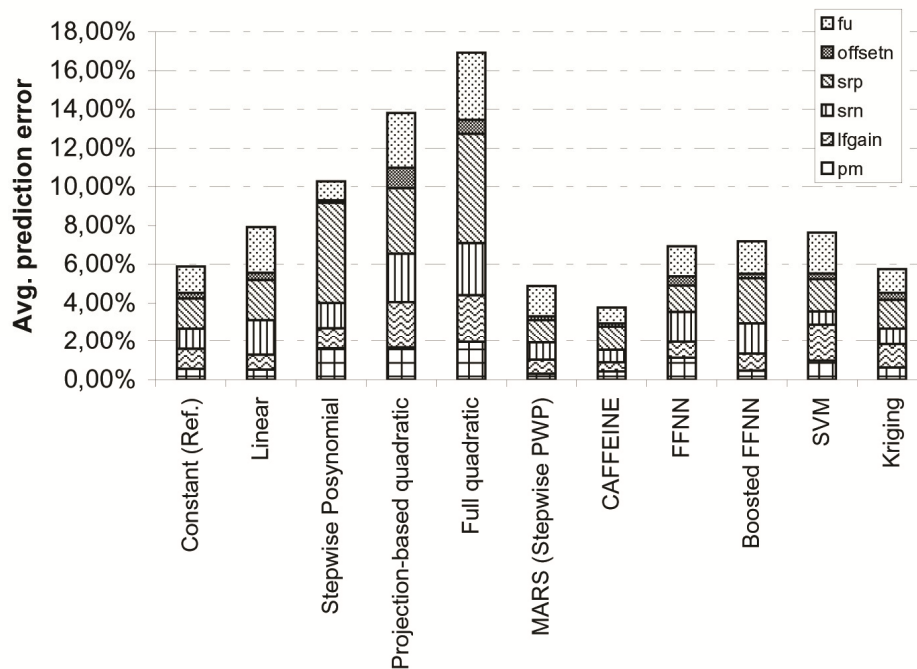


Figure 7: Comparison of prediction ability of CAFFEINE to state-of-the-art modeling techniques [27].

The results on different regressors inform us about the nature of the data and models. Progressing across the spectrum of polynomial complexity -- from the simplest linear models to posynomials to projection-based quadratic to full quadratic -- the prediction error continually worsens. It turns out that the polynomials even capture the *training* error poorly; for example the projection-based quadratic had a training error of about 10% for each performance. Since the prediction error became lower the more constrained the polynomial model was, this indicates that where the models do attempt to use the added flexibility to predict better, it backfires. In general, this is indicative that a polynomial functional template is not appropriate for circuit performance mappings, even for this relatively simple OTA circuit.

CAFFEINE only selects input variables that really matter. It is biased towards the axes of the input variables rather than being affine-invariant. That is, CAFFEINE expressions and search operators work on one or a few input variables at a time, as opposed to using all variables in a weighted sum. MARS did similarly, because its stepwise-forward nature makes it also biased towards the axes and is selective of input variables. While CAFFEINE had the best or near-best prediction error on 5 of the 6 performance goals, MARS had the best or near-best on 3. The other approaches loose prediction performance because they have different biases.

CAFFEINE has been extended to scale to higher-dimensional problems, *via* gradient-directed regularization to simultaneously prune basis functions and set coefficients for the remaining basis functions [6], a pre-evolution step to filter single-variable expressions [6], always considering all linear basis functions [6], and latent variables [29]. For details, we refer the reader to the cited references.

7. Other Applications

This section briefly describes other problem types that the CAFFEINE tool has been applied to: variation-aware modeling, behavioral modeling, and analytical tradeoff modeling.

CAFFEINE was applied to statistical modeling, to give insight into the mapping from design variables to process capability (Cpk) [26] for the 50-device circuit of **Fig. 8**. It followed the same methodology as performance modeling, except the Cpk is computed from SPICE simulation data having both local and global process variations. **Table 5** shows the extracted CAFFEINE equation, which has 6.3% testing error. Note that the technology variations are embedded in the numerical coefficients of the model -- Cpk is not a function of these process parameters, only their aggregate effect on the design variables. We see that just 5 variables are needed to get 6.3% test error: C_c , w_{dp2} , w_{dp1} , w_{mt4} , w_{mt1} . The variables comprise one compensation capacitor and four widths, and no lengths nor multipliers. There are significant nonlinear interactions among the variables. An increase to w_{mt4} will increase Cpk, as will a decrease to w_{mt1} . Cpk is quite dependent on the square root of C_c . Cpk can also be increased by increasing w_{dp2} (big effect) or increasing w_{dp1} (much smaller effect).

Table 5: CAFFEINE-Generated Equation of Cpk for 50-Device Circuit of Fig. 8.

$$Cpk = + 1231.4 + 4.21 \cdot 10^6 * w_{mt4}^2 / w_{mt1} - 0.0012 / \sqrt{C_c} \\ - 9.39 \cdot 10^8 * w_{dp2}^2 * \sqrt{w_{dp1}} * \min(0.104, 6.60 \cdot 10^7 - 76.9 / \sqrt{C_c}) \\ + 1.21 \cdot 10^{12} / \min(-4.96 \cdot 10^6, 10^{10} - 2.48 \cdot 10^5 / (\sqrt{w_{dp2}} * C_c))$$

An alternative to modeling process variation is for CAFFEINE to directly map process variables to performance. This approach was taken in [29], for problems having up to 340 process variables. Latent variable regression was used to aid scalability. **Table 6** is an example result for the bandwidth of the opamp circuit.

Table 6: CAFFEINE-Generated Equation of the BW as a Function of the Process Variables for the Circuit of Fig. 4.

$$BW = 1.184e+6 + 0.871 \cdot 10^6 * t_1 * \sqrt{\max(0, 5.214 * t_1)} + 0.213 \cdot 10^6 * t_1 \\ \text{where } t_1 = 1.338 \cdot 10^6 + 6.683 \cdot 10^3 * DPI.M2_{TOX} + (40 \text{ other linear terms})$$

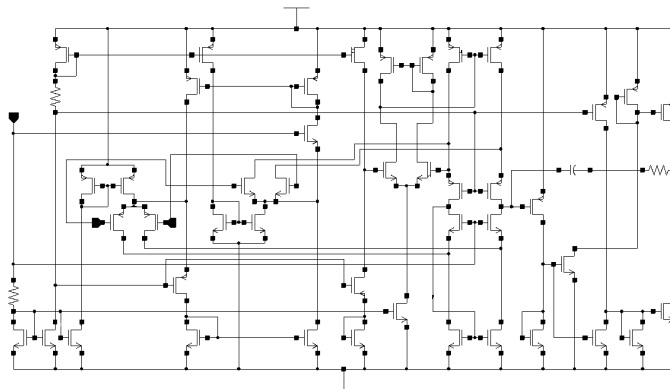


Figure 8: 50-transistor amplifier circuit.

CAFFEINE has also been used to extract behavioral models [28]. Despite much progress in automated behavioral modeling, manual design of models remains popular because humans can leverage their insights and vouch for the final model. CAFFEINE bridges manual and automated design by offering behavioral model “suggestions” to guide the modeling expert.

In [7] CAFFEINE has been used to extract whitebox models relating performance-tradeoff objectives for an amplifier. The input data set contained 1576 Pareto-optimal points in five objectives. One objective (gain bandwidth GBW) was set as the target output, and the other four became model input variables. **Table 7** shows the extracted model, having 4.1% training error.

Table 7: CAFFEINE-Generated Equation of BW as a Function of Other Performances.

$$GBW = 4.48 + 24.9 / \sqrt{A_{LF}} - (8.60 \cdot 10^6) / (A_{LF}^2 * \sqrt{SR})$$

8. Conclusion

This chapter has presented CAFFEINE, a tool to generate interpretable symbolic models of nonlinear analog circuit performances as a function of the circuit's design variables, without *a priori* requiring a model template. The keys to CAFFEINE are: a flow using SPICE simulation data, multi-objective GP search to extract template-free functions from the simulation data, and canonical-form constraints on the functions for interpretability (*via* a grammar). Visual inspection of the models demonstrates their interpretability. The CAFFEINE models also have lower average prediction error than several modern regression techniques. CAFFEINE has also been applied to variation-aware modeling, behavioral modeling, and tradeoff modeling.

Acknowledgments

The authors acknowledge the financial support of Solido Design Automation and FWO Flanders for parts of this work.

References

- [1] R.A. Rutenbar, G.G.E. Gielen, and J. Roychowdhury, “Hierarchical modeling, optimization and synthesis for system-level analog and RF designs”, in *Proc. of the IEEE*, Vol. 95, no. 3, pp. 640–669, March 2007.
- [2] G.G.E. Gielen, “Techniques and applications of symbolic analysis for analog integrated circuits: a tutorial overview”, in *Computer Aided Design of Analog Integrated Circuits And Systems*, R.A. Rutenbar, G.G.E. Gielen and B.A. Antao, Eds., IEEE, pp. 245–261, 2002.
- [3] G.G.E. Gielen, H. Walscharts, and W.M.C. Sansen, “ISAAC: A symbolic simulator for analog integrated circuits”, in *IEEE Journal of Solid-State Circuits*, Vol. 24, no. 6, pp. 1587–1597, December 1989.
- [4] A. Manthe, Z. Li, and C.-J. Richard Shi, “Symbolic analysis of analog circuits with hard nonlinearity”, in *Proc. of Design Automation Conference (DAC)*, pp. 542–545, 2003.
- [5] J. Yang, Z. Qi, and M. Gawecki, “Hierarchical symbolic piecewise-linear circuit analysis”, in *Proc. of the Behavioral Modeling and Simulation Conference (BMAS)*, pp. 140–145, 2005.
- [6] T. McConaghy and G.G.E. Gielen, “Template-free symbolic performance modeling of analog circuits *via* canonical form functions and genetic programming,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 28, no. 8, pp. 1162–1175, August 2009.
- [7] T. McConaghy, P. Palmers, G.G.E. Gielen, and M. Steyaert, “Automated extraction of expert knowledge in analog topology selection and sizing,” in *Proc. of the International Conference on Computer-Aided Design (ICCAD)*, San Jose, pp. 392–395, 2008.

- [8] A. Singhee and R.A. Rutenbar, "Beyond low-order statistical response surfaces: latent variable regression for efficient, highly nonlinear fitting," in *Proc. of the Design Automation Conference (DAC)*, pp. 256–261, 2007.
- [9] W. Daems, G.G.E. Gielen, and W.M.C. Sansen, "An efficient optimization-based technique to generate posynomial performance models for analog integrated circuits", in *Proc. of the Design Automation Conference (DAC)*, pp. 431–436, 2002.
- [10] X. Li, P. Gopalakrishnan, Y. Xu, and L. Pileggi, "Robust analog/RF circuit design with projection-based performance modeling", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 2–15, January 2007.
- [11] John R. Koza, *Genetic programming*. MIT Press, 1992.
- [12] P.A. Whigham, "Grammatically-based genetic programming", in *Workshop on Genetic Programming*, J.R. Rosca, Ed., 1995.
- [13] T. McConaghy, T. Eeckelaert, and G.G.E. Gielen, "CAFFEINE: template-free symbolic model generation of analog circuits via canonical form functions and genetic programming", in *Proc. of the Design Automation and Test Europe Conference (DATE)*, pp. 1082–1087, 2005.
- [14] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II", *IEEE Transactions on Evolutionary Computing*, Vol. 6, no. 2, pp. 182–197, August 2002.
- [15] D.C. Montgomery, *Design and analysis of experiments*, 6th edition, John Wiley & Sons, NY, USA. ISBN: 047148735X, 2004.
- [16] J.H. Friedman and B.E. Popescu, "Gradient directed regularization for linear regression and classification", in *Stanford University Department of Statistics, Technical Report*, February 2004.
- [17] T. Soule and R.B. Heckendorn, "An analysis of the causes of code growth in genetic programming", in *Genetic Programming and Evolvable Machines*, vol. 3, no. 3, pp. 283–309, September 2002.
- [18] E. Kirshenbaum and H.J. Suermondt, "Using genetic programming to obtain a closed-form approximation to a recursive function", in *Proc. of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 543–556, 2005.
- [19] P. Wambacq *et al.*, "Efficient symbolic computation of approximate small-signal characteristics", in *IEEE Journal of Solid-State Circuits*, Vol. 30, no. 3, pp. 327–330, March 1995.
- [20] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computing*, Vol. 3, no. 2, pp. 82–102, July 1999.
- [21] F. Leyn, G.G.E. Gielen, and W.M.C. Sansen, "An efficient DC root solving algorithm with guaranteed convergence for analog integrated CMOS circuits", in *Proc. of the International Conference on Computer-Aided Design (ICCAD)*, pp. 304–207, 1998.
- [22] N. Ampazis and S.J. Perantonis, "Two highly efficient second order algorithms for training feedforward networks," in *IEEE Transactions on Neural Networks*, vol. 13, no. 5, pp. 1064–1074, September 2002.
- [23] J.H. Friedman, "Multivariate adaptive regression splines", in *Annals of Statistics*, 19, pp. 1–141, March 1991.
- [24] J.A.K. Suykens and J. Vandewalle, *Least squares support vector machines*. World Scientific Publication Co., Singapore, 2002.
- [25] D.R. Jones, M. Schonlau, and W.J. Welch, "Efficient global optimization of expensive black-box functions", in *Journal of Global Optimization*, Vol. 13, no. 4, pp. 455–492, 1998.
- [26] National Institute of Standards and Technology, "What is process capability?" in *NIST / SEMATECH e-Handbook of Statistical Methods*, sec 6.1.6. <http://www.itl.nist.gov/div898/handbook/pmc/section1/pmc16.htm>
- [27] T. McConaghy and G.G.E. Gielen, "Analysis of simulation-driven numerical performance modeling techniques for application to analog circuit optimization", in *Proc. of the International Symposium on Circuits and Systems (ISCAS)*, pp. 1298–1391, 2005.
- [28] T. McConaghy and G.G.E. Gielen, "IBMG: Interpretable behavioral model generator for nonlinear analog circuits via canonical form functions and genetic programming", in *Proc. of the International Symposium on Circuits and Systems (ISCAS)*, pp. 5170–5173, 2005.
- [29] T. McConaghy, "Latent variable symbolic regression for high-dimensional inputs", *Genetic Programming Theory and Practice VI*, R. Riolo, U.-M. O'Reilly, and T. McConaghy, Eds., Springer, pp. 103–118, 2009.

CHAPTER 14**Symbolic Analysis Techniques for Fault Diagnosis and Automatic Design of Analog Circuits****Francesco Grasso, Antonio Luchetta^{*}, Stefano Manetti, Maria Cristina Piccirilli and Alberto Reatti***Università degli Studi di Firenze, Italy*

Abstract: This chapter is concerned with symbolic analysis techniques for fault diagnosis and automatic design of analog circuits. After a first paragraph describing a software tool for the symbolic analysis developed by the authors, it details testability and fault diagnosis of analog circuits and presents a symbolic approach to the design centering problem. In addition, it highlights modeling of power electronic circuits based on symbolic techniques.

Keywords: Analog circuits, Circuit simulation, Symbolic Simulation CAD, Testability, Ambiguity groups, Circuit faults, Electrical fault detection, Design centering, Design optimization, Yield optimization, Acceptability region, DC-DC power converters, Power converter modeling.

1. Introduction

During the last few years, symbolic analysis has been used as a useful tool to help the circuit designer during a wide variety of applications. Along the book, most of the main applications that make use of symbolic analysis in a more or less massive way are illustrated. This chapter treats the application of symbolic techniques to parametric fault diagnosis and to design centering of analog circuits, as well as the use of symbolic analysis as an operative help for the designer of DC/DC power converters. Furthermore in the chapter a symbolic simulation program developed by the authors during recent years is described: it constitutes the basic part of the software packages implementing the procedures developed for above-mentioned applications.

The Chapter is organized through three parts: the first one (Section 2) is an introduction to the program SAPWIN, developed by the authors; the second one presents the use of symbolic approach in two of the main tasks of analog circuit design: the fault diagnosis (Section 3) and the design centering (Section 4); finally the Section 5 contains an explanation, provided with examples, of the potential of symbolic analysis for DC/DC converter design.

2. The Symbolic Simulation Program SAPWIN

At present, few technical PC programs are available to perform the symbolic simulation of the electronic circuits. Moreover, several of the most famous general purpose commercial or open-source mathematical software packages do include symbolic computation capabilities. However they are usually neither easy to learn nor agile to use, moreover they are not optimized for the symbolic calculation of large systems and finally they do not have a standalone working capabilities. For these reasons, during the last few years the authors of this chapter have developed an

^{*}Address correspondence to Antonio Luchetta: Università degli Studi di Firenze, Italy; E-mail: antonio.luchetta@unifi.it

autonomous package for the symbolic simulation of the analog circuits, named SAPWIN [1-4] and at the present time arrived to the version 3.0 for Windows. The program is composed by two parts: a symbolic calculator engine and a schematic editor with graphical postprocessor.

2.1. Computational algorithms of SAPWIN

SAPWIN for Windows has now arrived to the version 3.0. It is completely coded in C++ language. Besides the natural evolution of the graphical interface that followed the increasing potentialities of PC platforms and compilers, significant changes have been made in the simulation engine routines. The ancestor of the program was a module written in LISP and C (at the beginning of '90 years) and called "SAPEC" (Symbolic Analysis Program for Electrical Circuits) which was able to calculate the transfer function of small circuits in the Laplace domain, using a permutation algorithm and with a very simple graphical interface (directly working in DOS). Subsequently, the first versions (1.0 and then 1.2) of SAPWIN [1] have been developed by introducing a completely new graphical interface devoted to Windows and substituting the permutation algorithm with another one based on a Laplace matrix-based recursive method, that drastically increased the speed of the analysis and the scale of analyzable circuits. Moreover approximation basic capabilities were introduced and several other features. The programming language became the C++ for all the modules. In the new millennium the program was rewritten another time, arriving to the version 3.0, still written in C++, with a further improvement of the graphical interface and with a new algorithm used for the symbolic analysis of the circuits. In the following a brief outline of the used algorithm and the way it works with input data are given. The description of the circuit is given by means of a SPICE-like netlist or directly within schematic editor of the program, which then provides to translate the scheme into an ASCII netlist. The present SAPWIN simulation engine is based on a two-graph method enhanced by authors, endowed with robustness and speed in comparison with matrix recursive techniques used in the former versions of the program. The starting point is a Two-graph Tree Enumeration method for the analysis of circuits containing only components with admittance representation and transconductances, shortly indicated as RC g_m circuits. After the construction of the voltage and current graphs G_V and G_I [5], the procedure can be implemented in one algorithm, in the following steps:

- a two-terminal component results in one branch situated between the same two nodes in both G_V and G_I ;
- a transconductance (g_m) results in one branch situated between control nodes in G_V and one branch situated between controlled nodes in G_I ; in both branches the weight of branch is given by transconductance value;
- the common trees can be determined by using, for instance, the efficient MRT algorithm [5];
- the determinant of the admittance matrix can be determined by the formula:

$$\Delta = \sum_{\substack{2\text{-graphs} \\ \text{com-trees}}} \varepsilon_i \left(\prod Y_{ict} \right)$$

where:

Y_{ict} = admittances situated over the i -th common tree;

$\varepsilon_i = \pm 1 = (\det m\{A_i\})(\det m\{A_v\})$, with $m\{A_i\}$ and $m\{A_v\}$ are the minors of the incidence matrices relating to considered trees;

- the network functions related to each pair of ports of the circuit can be determined using the formulas:

$$Z_m = \frac{V_1}{I_1} = \frac{\Delta_{11}}{\Delta}; \quad \frac{V_o}{I_m} = \frac{V_2}{I_1} = \frac{\Delta_{12}}{\Delta}; \quad \frac{V_o}{V_m} = \frac{V_2}{V_1} = \frac{\Delta_{12}}{\Delta_{11}}.$$

in a generic circuit with n nodes, where the input is assumed to be between nodes 1 and 0 and the output is assumed to be between nodes 2 and 0 and furthermore:

Δ is the determinant of the nodal admittance matrix;

Δ_{ij} is the ij -th cofactor of Δ .

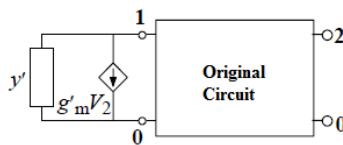


Figure 1: Circuit modified to obtain Δ_{11} , Δ_{12} and Δ .

A simple way for obtaining Δ_{11} , Δ_{12} and Δ is given by considering a slightly different circuit with respect to the original one, containing and extra admittance y' and a voltage controlled current source $g'_m \cdot V_2$ (**Fig. 1**), and calculating the determinant of the admittance matrix for this new circuit:

$$\Delta' = \Delta + y' \Delta_{11} + g'_m \Delta_{12}$$

In this way all the network functions can be evaluated.

The presence of non-transconductance controlled sources can be solved by introducing nullator and norator concepts. Nullator and norator are a couple of two-terminal ideal components. Working together they form a two-port component called “nullor”, that represents a suitable model for an ideal operational amplifier. Relations of these components are in **Fig. 2**. See also the Chapter 3 of this same book for them and a more extended treatment.

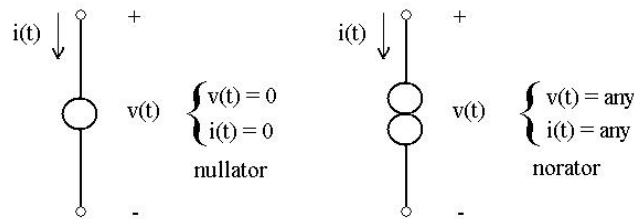


Figure 2: Norator and nullator symbols.

These two ideal components are easily manageable, taking into account that for the nullator the two terminal nodes do collapse in only one in G_V and for the norator the two terminal nodes do collapse in only one in G_I ; this would create problem in tree enumeration, but it can be overcome leaving the nodes separated, but forcing the correspondent branch to belong to the tree (*i.e.* the nullator branch on G_V and norator

branch on G_1) [6]. Finally the equivalent circuits in **Fig. 3** can be used to model all the non-transconductance generators using nullor concept.

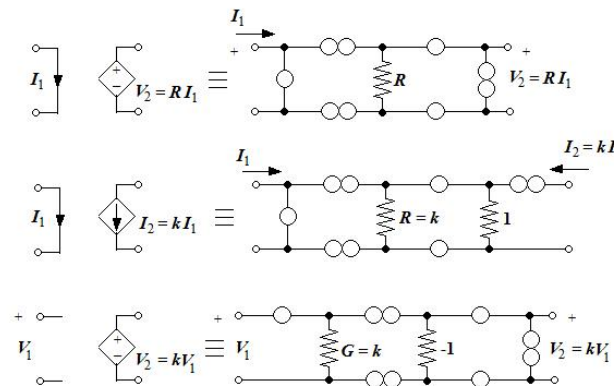


Figure 3: Equivalent circuits for non-transconductance controlled generators.

By implementing this algorithm SAPWIN program is able to handle any kind and size of circuit.

2.2. Program main capabilities

In **Fig. 4** a general screenshot of the program SAPWIN is given. It includes in a single main window the possibility of drawing one or more circuit schematics, simulating the circuit(s), viewing and evaluating the results, eventually changing parameter values also after elaboration. The output can be evaluated also in the phasor domain. In the next sub-sections a more detailed description of the various program commands are given.

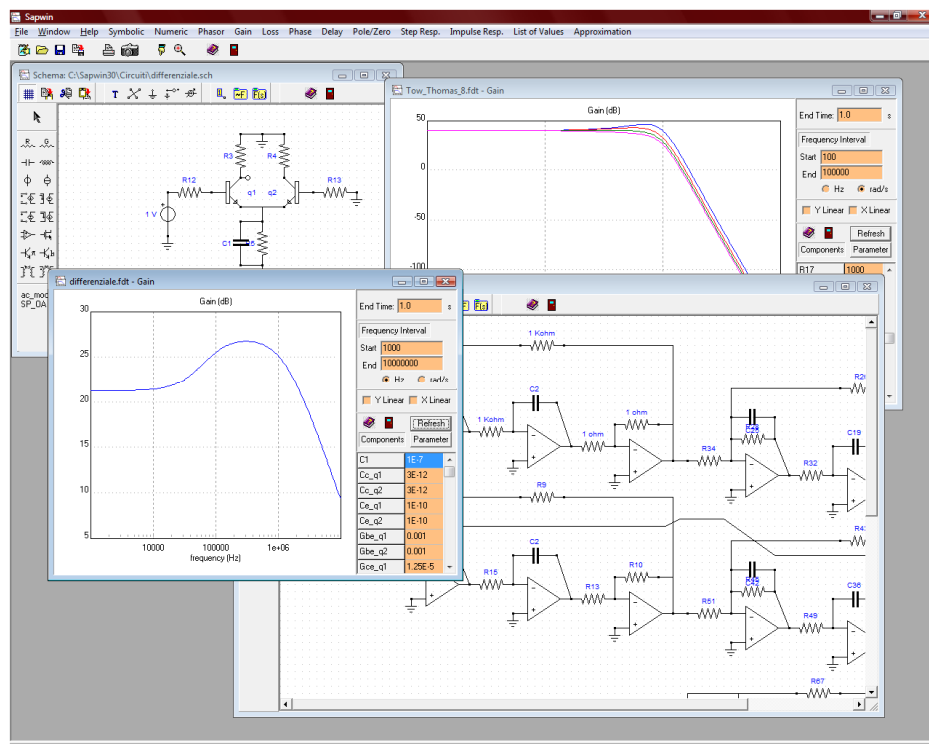


Figure 4: Main screen of SAPWIN program.

2.2.1. Schematic entry

The schematic entry, that is clearly visible also in **Fig. 4**, can be drawn over an initially blank table, in the same way of several other recent schematic editors of analog or digital circuits. All the standard two-terminal or two-port elements can be taken from a component list positioned on the left of the editor window. The available devices are the two-terminal passive fundamental ones: resistor (both in resistance R or conductance G representation), inductor L , capacitor C ; the independent ideal voltage and current sources; the four type of controlled sources: the voltage dependent sources, controlled in voltage VCVS (associated symbol E) and in current CCVS (associated symbol Y) and the current dependent sources, controlled in voltage VCCS (associated symbol H) and in current CCCS (associated symbol F). The passive two-port are the ideal transformer and the mutual inductance. The active two-ports are the ideal operational amplifier, the MOSFET and the BJT. The inclusion of a transistor device entails the expansion of the component into the equivalent circuit; in BJT case the equivalent circuit can be a hybrid parameter model or a π -model (Giacoletto equivalent); in MOSFET case the equivalent circuit is the g_m -model with the eventual presence of parasitic capacitances (selectable in the initial mask). The connection among the elements should be completed by means of the “wire” tool, which simulate an equipotential path (an ideal conductor wire). It is essential to underline that, when a wire crosses another wire, the editor will put a new connection (a new node) in the intersection point if and only if the two wires are orthogonal, otherwise, if between the two lines there is not a right angle, that point is considered a wire-bridge (and no new node is added). This difference is shown in **Fig. 5**.

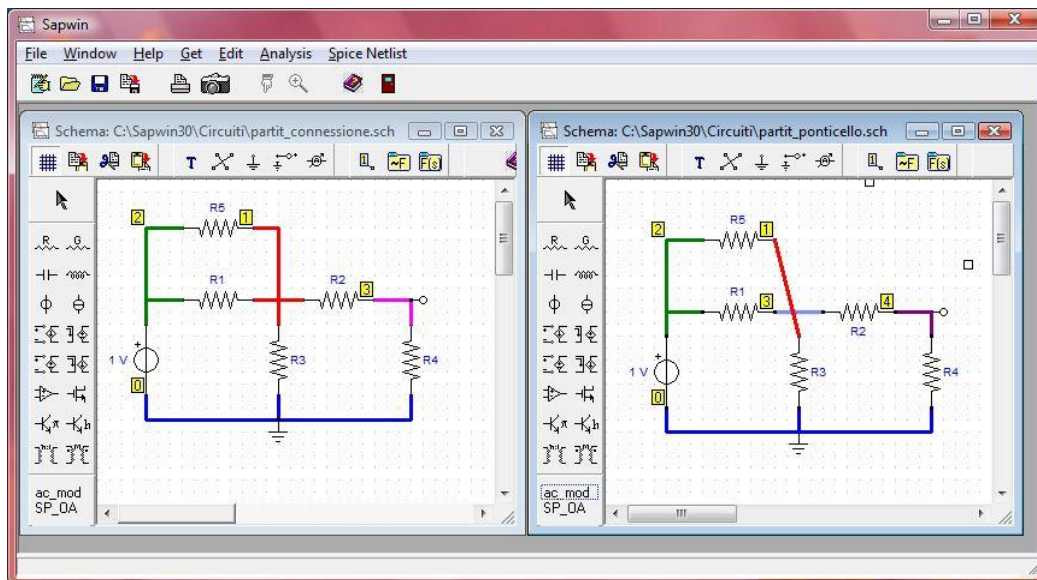




Figure 5: Schematics of two similar circuits. In the left one the wires are orthogonal and connected in the red node [1], in the right one the wires are crossing in a not right angle and they generate two distinct nodes [1] and [3].

The schematic of the circuit must be provided with at least one ground connection (multiple ground connections can be inserted, yielding equipotential the nodes which are connected to). The circuit under simulation must also include one output probe in order to set up the simulation. The output can be a voltage probe attached to a pin (the voltage will be calculated with respect to the ground “reference” node) or a current probe along a wire (in this case an appropriate space must be left to position the

current probe). The schematic can be saved in a file with a name chosen by the user and extension .sch.

In the same editor it is possible to create and save sub-circuit schemes (sometimes called "user defined models"). A sub-circuit is a two-port element that includes a group of components. It can be re-used several times without redrawing the whole group. A sub-circuit can be drawn just as a full circuit, taking into account of the following differences:

- it cannot contain independent sources, output current, other subcircuits;
- the accessible terminals of the subcircuits are obtained by placing output voltages;
- the number assigned to each terminal is the order in which they have been drawn;
- the side in which each terminal will appear in the block that will represent the subcircuit depends on the orientation of the placed terminal (o—• left side, •—o right side,  top side,  bottom side).

The numbers assigned to terminals can be checked and they are shown in green. Each component of a subcircuit can be made symbolic or not, editing the component properties. The symbolic components will appear in the symbolic network functions suffixed by the subcircuit name, for example: a symbolic resistor Ra of the subcircuit X5 will appear in the symbolic network function as Ra_X5.

In designing regular circuits, the available sub-circuits are reported in a list under the component buttons.

2.2.2. *Circuit simulation*

When the circuit to simulate has been drawn, the simulation can start. Before starting, it is eventually possible to check the number and the position of the nodes with the command button situated in the trace bar of the schematic editor. In this way it is also possible to check if the connection points are just those we want to be, and if the number of nodes is the expected one. When the schematic of the circuit is saved (or the first time that simulation is launched) the program generates also the netlist of the circuit to be analyzed and it saves such netlist in a file with the same name of schematic and extension .crc. This is a SPICE-like netlist that contains the following items:

- The first row is the name of the circuit schematic (including the path).
- Each row (starting from the second one) represents a component. The format is "*Sname n1 n2 n3 n4 v s*", where:
 - *Sname* is the name of the component; **S** is the symbol related to the specific two-terminal or two-port element (R for resistor, C for capacitor, etc.) and *name* is the suffix which identifies the given component (it can be composed of letters, numbers and underscore sign, for example R1, Cg1, L_sa, etc.). If the list is generated from the schematic editor, it coincides with the name inserted in the schematic dialog box.

- $n1$ $n2$ $n3$ $n4$ are the four (at the most) integer values that identify the nodes where the component is connected. If it is a two terminal component only $n1$ and $n2$ are present. If the two terminal component is a voltage source, then $n1$ identifies the positive polarity; if the two terminal component is a current source, then $n1$ identifies the outgoing current point; if the element is a two-port element, then $n1$ and $n2$ are the output port nodes and $n3$ and $n4$ are the input port nodes; but now the current convention is opposite with respect to one-port case, *i.e.* the first of the two terminals is relative to the entering current. For the operational amplifier element, which in the schematic has only 3 terminal pins, the $n4$ node is always put equal to 0 (ground node).
- v is the decimal number which represents the nominal value of the component. It is not present for operational amplifier case.
- s is a boolean number (0 or 1) which is equal to 0 if the component is symbolic, 1 if it is included as a numeric value into the calculation.
- The list must include the following command lines:
 - `.OUT n` or `.IOUT n1 n2`. The first one indicates to the simulator that the output variable is the voltage on the node n . The second one indicates to the simulator that the output variable is the current that flows from the node $n1$ toward the node $n2$.
 - `.END` establishes the end of the netlist read by the simulator.

An example of a simple circuit and its netlist are given in **Fig. 6**. The generated netlist is passed to the program module SAPEC.exe, which elaborates it in order to start the simulation, which runs as described in the previous section.

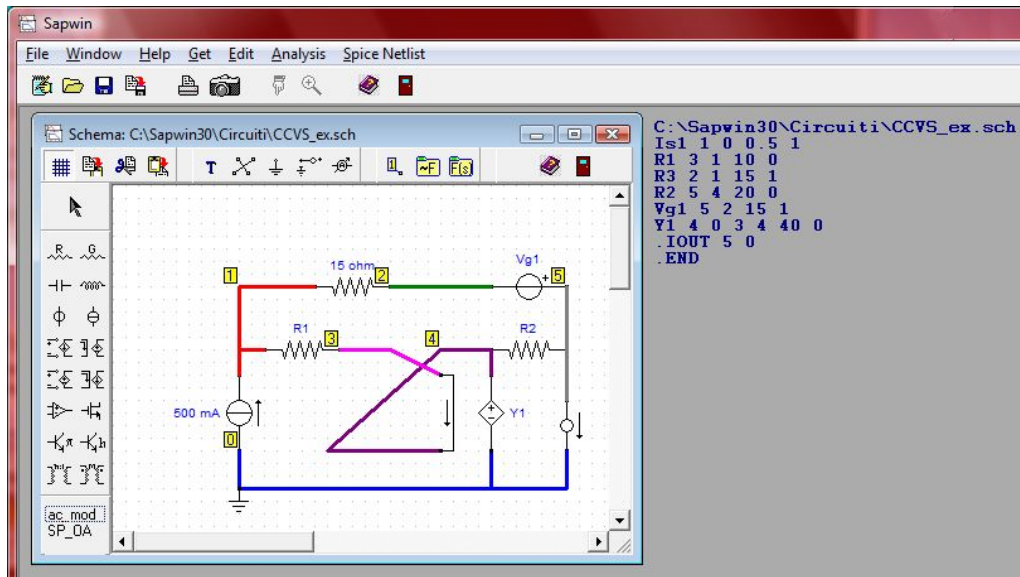


Figure 6: A circuit schematic and the relating netlist.

During this phase a window appears to report some information about the circuit and about the simulation progress, as shown in **Fig. 7**.

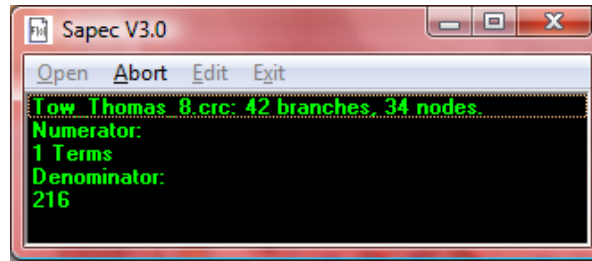


Figure 7: Simulation window.

2.2.3. Approximation capabilities

On the other hand it should be observed that symbolic analysis of a circuit, also for medium sized system, does result in a very large output expression, that could be unmanageable or hardly readable. In order to improve the manageability of generated functions, approximation methods can be used. SAPWIN is able to approximate the output functions, choosing a given error over a range of frequencies and evaluating it after the generation with an efficient approximation after generation method. This feature will be anyway improved in the future works.

2.2.4. Output files

The simulation generates two different output files. The first one is an internally used output file, with the same filename of the circuit and .fdt file extension. This is a binary file that contains the final expression in form of binary coded structures, each one representing one term of numerator and denominator. The data stored in the files are:

- the total number of structures;
- for each structure:
 - degree of the expression
 - numerical part of the expression
 - number of symbolic components
 - group of strings (one for each symbolic component) terminated by a '\0' character

This file can be easily and quickly accessed by program functions in order to extract in any moment the requested data.

The second output file is an ASCII text file, with the same filename of the circuit and .txt file extension. It contains the resulting expression of the simulation in the Laplace domain, reported as a list of terms, each one positioned on a different row in increasing power of s , for numerator and denominator, as in the following :

```
( - EAo C1 R2 Roi + C1 R2 Roi + C1 R2 R3) s
( - EAo C2 C1 R2 R3 Roi + C2 C1 R2 R3 Roi) s^2
-----
( - EAo R2 Roi - EAo R1 Roi + R2 Roi + R2 R3 + R1 Roi + R1 R3)
( - EAo C1 R1 R2 Roi + C1 R1 R2 Roi + C1 R1 R2 R3 - EAo C2 R1 R2 Roi + C2 R2 R3 Roi
+ C2 R1 R3 Roi + C2 R1 R2 Roi + C2 R1 R2 R3) s
( - EAo C2 C1 R1 R2 R3 Roi + C2 C1 R1 R2 R3 Roi) s^2
```

The output format can be easily parsed for further re-elaborations [7].

2.3. SAPWIN output presentation

As soon as the simulation ends, an output window is open and the symbolic expression is presented. At the right side of the output window the list of parameters with values is present. On the top of right side the end time for the time response and the frequency interval for the frequency response can be inserted (they are visible in **Fig. 8**). The resulting expression can be viewed both in symbolic or in numerical form (in this last case the values are those assigned in the schematic editor and now appearing in the parameter edit boxes).

2.3.1. Graphical tools

The output of the simulation engine can be elaborated also in a graphical way, to obtain seven different diagrams:

- *Magnitude response*: it is the diagram of the module of the network function, drawn in relation to the frequency (or angular frequency). Linear or logarithmic scale can be used (in the last case it is usually called “Gain”);
- *Loss*: just the inverse of gain (in some test or simulation it is more significant);
- *Phase response*: it is the diagram of the phase (in degrees) of the network function, drawn in relation to the frequency (or angular frequency);
- *Time Delay*: it is the time delay of the response;
- *Poles/zeros*: it shows the poles and the zeroes of the network function in the complex plane;
- *Step response*: it is the step response of the circuit;
- *Impulse response*: it is the impulse response of the circuit.

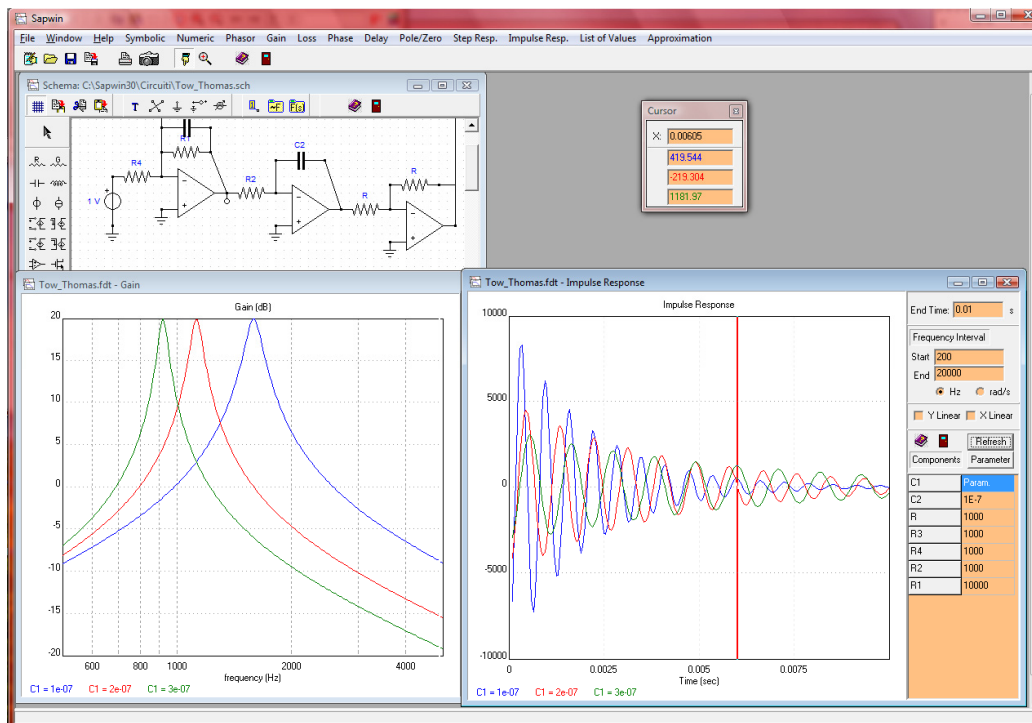


Figure 8: Window containing the diagrams of the gain and impulsive response of an active circuit parameterized with respect to one capacitive component.

Algorithms for poles/zeros calculation and for the inversion of the network function to obtain step and impulse response can be found in [8].

When a diagram is visualized in the window, it is also possible to add an X-Y cursor to evaluate the numerical values on the diagram.

2.3.2. Phasor calculation

The result of simulation is given in the Laplace domain. It is also possible to evaluate it as a phasor at a given frequency (or angular frequency). The phasor of the response is shown both in real and imaginary part and in amplitude and phase, at the frequency selected in the right side window. For this kind of output, in the list of component values it is possible to specify complex values for the symbolic independent sources, in the form of amplitude and phase (in degrees), by using the letter ‘P’ between the amplitude and the phase; for instance, to assign to an independent voltage source a value of 100V in amplitude and 45° in phase, the right form is: “100 P 45”. The feature is particularly useful for circuits with multiple inputs. To utilize this opportunity, the independent sources must be put in symbolic form in the schematic editor, by editing such components before starting the analysis.

3. Symbolic Approach to the Fault Diagnosis Problem

This section deals with fault diagnosis of analog circuits, the first of the three application fields of symbolic analysis considered in this chapter.

The symbolic approach can be very useful for developing efficient methodologies of parametric fault diagnosis. A parametric (soft) fault is a fault that can be described as a parameter variation and does not influence the circuit topology. The parametric fault diagnosis can be viewed as a problem in which, given a circuit structure and some I/O relations, the component values have to be determined. In other words, it can be considered as a parameter identification problem. Actually, it is usually sufficient to determine which parameters are faulty, *i.e.* which parameters are out of tolerance, without determining the values of all the components. In any case, the symbolic approach is a natural choice in this kind of problems, because a I/O relation, in which the component values are the unknowns, is properly represented in symbolic form.

The I/O relations used in parametric fault diagnosis are constituted by a set of equations nonlinear with respect to the unknown component values. These equations are related to a set of measurements carried out on specific points of the circuit, called test points. The test point selection is a non trivial operation, because not all the possible test points can be easily accessed. For example, it is usually very difficult to measure currents without breaking connections. In other words, the test point selection must take into account practical measurement problems strictly tied with the used technology and with the application field of the circuit. So, in order to perform a good test point selection, it is necessary to have a quantitative index for comparing the different possible choices. The testability measure concept meets this requirement. Following what above stated, fault diagnosis can be split into two different steps: testability analysis and fault location. The testability analysis consists in the testability evaluation and in the ambiguity group determination, the fault location consists in the determination of the faulty components. In the following, these two steps are considered for analog, linear, time-invariant circuits, highlighting the advantages of symbolic analysis.

3.1. Testability and ambiguity groups

Testability concept is strictly tied to that of network-element-value-solvability, which was first introduced by Berkowitz [9]. Successively, a very useful testability measure was introduced by Saeks *et al.* [10-13]. Other definitions have been presented in subsequent years (see, *e.g.*, [14-16]), and, then, there is not a universal definition of analog testability. However, the Saeks' definition has been the most widely used [17-20], because it provides a well-defined quantitative measure of testability. In fact, once a test point set has been selected, by representing the circuit under test through a set of equations nonlinear with respect to the component parameters, this testability definition gives a measure of solvability of these equations and indicates the ambiguity resulting from an attempt to solve such equations in a neighborhood of almost any failure. Therefore, this testability measure allows to know a priori if a unique solution of the fault diagnosis problem exists. Furthermore, if this solution does not exist, it gives a quantitative measure of how far we are from it, *i.e.*, how many components cannot be diagnosed with the given test point set.

When testability is low, an important concept is that of ambiguity group. An ambiguity group is, essentially, a group of components where, in case of fault, it is not possible to uniquely identify the faulty one. A canonical ambiguity group is a "minimal" ambiguity group, *i.e.*, a group that does not contain, within it, ambiguity groups of lower order. The canonical ambiguity groups give information about the solvability of the fault diagnosis problem with respect to each component, in case of bounded number of faults (k -fault hypothesis) [21].

Summarizing, once a set of test points has been selected, independently of the method effectively used in the fault location phase, the testability measure gives a theoretical and rigorous upper limit to the degree of solvability of fault diagnosis problem at a global level, while the ambiguity group determination gives the solvability degree at a component level. If these important concepts are not properly taken into account, the quality of the obtained results is severely limited [22]. So, testability analysis is essential to both the designer, who must know which test points to make accessible, and the test engineer, who must know how many and what parameters can be uniquely isolated by the planned tests.

The first algorithms for evaluating testability measure as previously defined were developed by using a numerical approach [23, 24]. However these methods were suitable only for networks of moderate size, because of the inevitable round off errors introduced by numerical algorithms, which render the obtained testability only an estimate. This limitation has been overcome with the introduction of the symbolic approach [25-30] through an efficient manipulation of algebraic expressions [31-33]. Using testability evaluation algorithms it is not difficult to realize procedures for canonical ambiguity group determination [29, 30, 34]. In the following an efficient technique for testability evaluation [34, 35] is reported. It permits also an easy determination of the ambiguity groups.

Referring to parametric fault diagnosis, the testability measure T is given by the maximum number of linearly independent columns of the Jacobian matrix associated with the fault diagnosis equations. The solution of these equations can be split in two phases. In the first phase, starting from the measurements carried out on the selected test points at different frequencies, the coefficients of the fault diagnosis equations are evaluated [35]. In the second phase, the circuit parameter values are obtained by

solving the nonlinear system constituted by the equations expressing the previously determined coefficients as functions of the circuit parameters. In this way, by considering the K fault diagnosis equations:

$$h_l(\mathbf{p}, s) = \frac{N_l(\mathbf{p}, s)}{D(\mathbf{p}, s)} = \frac{\sum_{i=0}^{n_l} \frac{a_i^{(l)}(\mathbf{p})}{b_m(\mathbf{p})} \cdot s^i}{s^m + \sum_{j=0}^{m-1} \frac{b_j(\mathbf{p})}{b_m(\mathbf{p})} \cdot s^j} \quad l = 1, \dots, K \quad (1)$$

the following nonlinear system has to be solved:

$$\begin{cases} \frac{a_0^{(1)}(\mathbf{p})}{b_m(\mathbf{p})} = A_0^{(1)} & \dots & \frac{a_{n_1}^{(1)}(\mathbf{p})}{b_m(\mathbf{p})} = A_{n_1}^{(1)} \\ \vdots & & \\ \frac{a_0^{(K)}(\mathbf{p})}{b_m(\mathbf{p})} = A_0^{(K)} & \dots & \frac{a_{n_K}^{(K)}(\mathbf{p})}{b_m(\mathbf{p})} = A_{n_K}^{(K)} \\ \frac{b_0(\mathbf{p})}{b_m(\mathbf{p})} = B_0 & \dots & \frac{b_{m-1}(\mathbf{p})}{b_m(\mathbf{p})} = B_{m-1} \end{cases} \quad (2)$$

where $\mathbf{p} = [p_1, p_2, \dots, p_R]^t$ is the vector of the potentially faulty parameters, $A_i^{(l)}$ and B_j ($i=0, \dots, n_l, j=0, \dots, m-1$) are the coefficients of the fault diagnosis equations in (1) which have been calculated in the previous phase. The Jacobian matrix \mathbf{B}_C of this system can be considered as the testability matrix:

$$\mathbf{B}_C = \begin{pmatrix} \frac{\partial \frac{a_0^{(1)}}{b_m}}{\partial p_1} & \frac{\partial \frac{a_0^{(1)}}{b_m}}{\partial p_2} & \dots & \frac{\partial \frac{a_0^{(1)}}{b_m}}{\partial p_R} \\ \dots & \dots & \dots & \dots \\ \frac{\partial \frac{a_{n_1}^{(1)}}{b_m}}{\partial p_1} & \frac{\partial \frac{a_{n_1}^{(1)}}{b_m}}{\partial p_2} & \dots & \frac{\partial \frac{a_{n_1}^{(1)}}{b_m}}{\partial p_R} \\ \dots & \dots & \dots & \dots \\ \frac{\partial \frac{a_0^{(K)}}{b_m}}{\partial p_1} & \frac{\partial \frac{a_0^{(K)}}{b_m}}{\partial p_2} & \dots & \frac{\partial \frac{a_0^{(K)}}{b_m}}{\partial p_R} \\ \dots & \dots & \dots & \dots \\ \frac{\partial \frac{a_{n_K}^{(K)}}{b_m}}{\partial p_1} & \frac{\partial \frac{a_{n_K}^{(K)}}{b_m}}{\partial p_2} & \dots & \frac{\partial \frac{a_{n_K}^{(K)}}{b_m}}{\partial p_R} \\ \dots & \dots & \dots & \dots \\ \frac{\partial \frac{b_0}{b_m}}{\partial p_1} & \frac{\partial \frac{b_0}{b_m}}{\partial p_2} & \dots & \frac{\partial \frac{b_0}{b_m}}{\partial p_R} \\ \dots & \dots & \dots & \dots \\ \frac{\partial \frac{b_{m-1}}{b_m}}{\partial p_1} & \frac{\partial \frac{b_{m-1}}{b_m}}{\partial p_2} & \dots & \frac{\partial \frac{b_{m-1}}{b_m}}{\partial p_R} \end{pmatrix} \quad (3)$$

Independently of the used fault location method, the testability value $T = \text{rank} \mathbf{B}_C$ gives information on the solvability degree of the problem, as explained below:

- if $T = R$ (R is the number of unknown elements), the parameter values can be theoretically uniquely determined starting from a set of measurements carried out on the test points;
- if $T < R$, a locally unique solution can be determined only if $R - T$ components are considered not faulty.

Generally T is not maximum and the hypothesis of a bounded number k of faulty elements is made (k -fault hypothesis), where $k \leq T$. Then, the testability gives the solvability degree of the fault diagnosis problem and, consequently, the maximum possible k .

The matrix \mathbf{B}_C gives other information besides the global solvability degree of the fault diagnosis problem. In fact, by observing that each column is relevant to a specific parameter of the circuit and by considering the linearly dependent columns of \mathbf{B}_C , other information can be obtained. For example, if a column is linearly dependent with respect to another one, this means that a variation of the corresponding parameter provides a variation on the fault equation coefficients indistinguishable with respect to that produced by the variation of the parameter corresponding to the other column. This means that the two parameters are not testable and they constitute an ambiguity group of the second order. By extending this reasoning to groups of linearly dependent columns of \mathbf{B}_C , ambiguity groups of higher order can be found. In the case of low testability and k -fault hypothesis, whatever fault location method is used, it is necessary to be able to select as potentially faulty parameters a set of elements that represents as best as possible all the circuit components. To this end, the determination of both the canonical ambiguity groups and surely testable group is of fundamental importance. As reported in [21], a set of k parameters constitutes a canonical ambiguity group of order k if the corresponding k columns of the testability matrix \mathbf{B}_C are linearly dependent and every sub-set of this group of columns is constituted by linearly independent columns, while a set of n parameters, whose corresponding columns in the testability matrix \mathbf{B}_C do not belong to any ambiguity group, constitutes a surely testable group of order n .

In order to understand the importance of canonical ambiguity groups and surely testable group, consider the Sallen-Key band-pass filter, shown in **Fig. 9**, where V_o is the test point. The program SYFAD (SYmbolic FAult Diagnosis) [34, 35], based on the software package SAPWIN, is able to yield both testability and canonical ambiguity groups. In **Fig.10** the program results are shown. As it can be seen, there are two canonical ambiguity groups without elements in common. The first group is of the second order and, then, it is not possible to select a set of components giving a unique solution. The surely testable group is constituted by G1 and C1. As the testability is equal to three, we can take into account at most a 3-fault hypothesis, *i.e.*, a possible solution can be obtained if only three component values are considered as unknowns. The elements to be selected as representative of the circuit components are the surely testable group components and only another component belonging to one of the two canonical ambiguity groups. This group of components is defined testable. Suppose, for example, a single fault case. Independently of the used fault location method, if the obtained solution gives as faulty element C1 or G1, the fault can be localized with certainty, because both C1 and G1 belong to the surely testable group. If a component belonging to the second order canonical ambiguity group is localized as potentially faulty element, it is only possible to know that there is a fault in this ambiguity group, but it cannot be exactly located, because there is not a unique solution. Instead, if a component belonging to the third order ambiguity group is located as faulty element, there is a unique solution, and then the fault can be located

with certainty. In other words, a fault in a component of this group can be counterbalanced only by simultaneous faults on all the other components of the same group. However, by the hypothesis of single fault, this situation cannot occur.

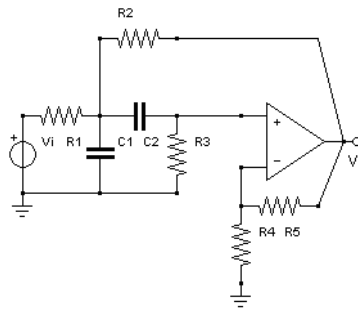


Figure 9: Sallen-Key band-pass filter.

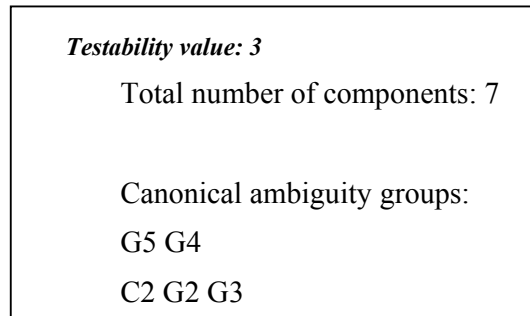


Figure 10: Program results for the circuit in Fig. 9.

From the example, it is possible to understand that a testable group is a group of potentially faulty components giving solution to the problem of fault location. It permits to confine the presence of faults to well-defined groups of components [21]. A testable group is easily obtainable through a combinatorial procedure, starting from the canonical ambiguity group knowledge.

One of the first algorithms for ambiguity group determination has been presented in [20]. In [34, 35] a combinatorial method, based on a symbolic approach, has been implemented in the program SYFAD. In the procedure a classical total pivot method is used on \mathbf{B}_C and its submatrices. Successively, another efficient numerical procedure for ambiguity group determination, based on the QR factorization of the testability matrix, has been proposed in [36, 37]. Nevertheless, this last method, even if not combinatorial, results very complex in the search of canonical ambiguity groups. Furthermore, although the QR decomposition approach presents several interesting features, it suffers from problems related to round-off errors, specially if the matrix is rank deficient, so that the numerical rank obtained is, often, only an estimate of the effective rank. These numerical problems are mostly overcome by the use of the Singular-Value Decomposition (SVD), which allows to obtain the effective numerical rank of the matrix, taking into account round-off errors [38]. So, by exploiting the great numerical robustness of the SVD, an accurate evaluation of testability value and an efficient procedure for canonical ambiguity group determination can be obtained [29]. The program TAGA (Testability and Ambiguity Group Analysis), based on the software package SAPWIN, yields testability and canonical ambiguity groups exploiting the SVD

[29]. It is important to remark that the availability of network functions in symbolic form strongly reduces the computational effort in the determination of the \mathbf{B}_C matrix entries, because they can be simply led back to derivatives of sums of products. The testability and ambiguity group determination can be performed by assigning arbitrary values to the components, because testability does not depend on component values [11].

3.2. Fault location

Analog fault diagnosis procedures [17, 39, 40] are usually classified in two basic groups: Simulation Before Test (SBT) techniques and Simulation After Test (SAT) techniques. The SBT techniques are based on generated off line fault dictionaries. Usually, SBT is suitable for single catastrophic fault location because of the very large dictionary size in multiple soft fault situations. The SAT approaches are suitable to diagnose parametric faults (*i.e.* deviations of parameter values from a given tolerance). In these methods, starting from the measurements carried out on the selected test points, the network parameters are estimated and compared to those of the fault-free network for identifying the fault. The use of symbolic methods is particularly suited for SAT techniques and, in particular, for those based on parameter identification. This is due to the fact that SAT approaches need more computational time than SBT approaches. Using a symbolic approach, noteworthy advantages can be reached, not only in computational terms, but also because it automatically includes the testability analysis in the fault diagnosis procedure.

Symbolic techniques are used in parametric fault diagnosis methods based on frequency domain measurements and on the k -fault hypothesis. The fault location techniques can be substantially split in two groups: techniques based on a bilinear decomposition of the fault equations and techniques based on a Newton-Raphson approach. The techniques based on a bilinear decomposition of the fault equations [41-44] are usually suitable for single and double fault cases, because they become excessively complex for a greater fault hypothesis. However, this is not a so big limitation, since, in practical circuits, the single fault case is the most frequent, the double fault case is less frequent and the case of all faulty components is almost impossible. The Newton-Raphson based techniques are generally suitable for any possible fault hypothesis, as, for example, it happens for the technique reported in [35]. In this technique the fault diagnosis equations are expressed as in (1) and the parameter evaluation is led back to the solution of the nonlinear system (2), whose testability matrix is reported in (3). By indicating with R the total number of circuit parameters, with k the number of potentially faulty parameters and with T the testability value ($T \leq R$ and $k \leq T$), the fundamental steps of the fault diagnosis procedure can be so summarized:

1. evaluation of T ;
2. determination of all the possible combinations of k testable parameters;
3. application of the Newton-Raphson method to each testable group of k parameters.

A group of k elements is testable if the related columns of \mathbf{B}_C are linearly independent. To know if a group of k elements is testable, the submatrix of \mathbf{B}_C constituted by the columns related to the selected parameters must be triangularized. If the chosen k elements are testable, the first k rows of the matrix represent the independent equations. So, a nonlinear system with k -equations and k -unknowns is obtained. It can be solved with the classical Newton-Raphson method, by assigning the nominal values to the other $R-k$ parameters. As well known, in the Newton-Raphson method it is necessary to evaluate the Jacobian matrix that, in this case, is a

submatrix of the testability matrix \mathbf{B}_C . To assure convergence to the Newton-Raphson procedure, the starting point has to be chosen close enough to the solution. The way to solve this and other problems, such as the effect of components tolerance, is shown in [35]. Each solution obtained with the Newton-Raphson method gives a possible set of faulty components. Multiple solutions can exist, due to the system nonlinearity and to the fact that the system is solved with respect to any possible group of k testable components, each one indicating a different possible fault situation, *i.e.* a different out of tolerance parameter group. Also this problem can be overcome by using a procedure of reduction of the solutions [35]. In the automation of the fault diagnosis procedure, implemented in the software program SYFAD, the availability of the network functions in completely symbolic form permits to simplify not only the testability analysis, but also the repeated solution of the system with different combinations of potentially faulty parameters. In fact, the Jacobian matrices, relevant to the application of the Newton-Raphson method to each testable component combination, are simply submatrices of the testability matrix.

The symbolic approach gives noteworthy advantages not only in the phases of testability analysis and solution of the fault diagnosis equations, but also in the search of the best measurements frequencies. In fact, this choice influences the fault location, because the solution of the fault diagnosis equations is perturbed by measurement errors and component tolerances. The choice of a suitable set of measurement frequencies allows to minimize the effect of these perturbations. The use of symbolic techniques can result very useful in solving this problem. In [45-48] some procedures for selecting the set of frequencies which leads to a good location of parametric faults in analog linear circuits are reported.

A detailed overview on the procedures of fault location based on the use of symbolic techniques is reported in [49], where additional information on testability analysis and on the test frequency selection is given. [49] presents also a brief discussion on the application of symbolic techniques to nonlinear analog circuits. Furthermore, a fault diagnosis technique applicable to nonlinear analog circuits is summarized [50, 51].

Finally, it is important to remark that the most important results achieved at today in the application of the symbolic techniques to fault diagnosis are relevant to the testability analysis, where the symbolic approach gives excellent results. For what concerns the phase of fault location, the symbolic approach is not the only possible one [52-54]; for example, good results have been obtained also by using neural networks or genetic algorithms. However, being testability analysis the initial step in the analog fault diagnosis, necessary for any kind of fault location procedure, the symbolic approach is indeed very useful and can give a noteworthy contribution for reducing the gap between analog and digital fields.

4. Symbolic Approach to the Design Centering Problem

One of the aims of analog design is to determine a circuit topology and the component parameter values so to maximize the manufacturing yield. For this reason, the nominal values of the circuit parameters are selected in such a way to ensure that the behavior of the circuit remains within specifications with the greatest probability. This phase of analog design is that of design centering.

The problem of design centering attempts to ensure that, under circuit parameter value changes, the circuit response remains within the performance specifications for

the whole range of environmental conditions, maximizing the probability that the specifications are met after the parametric variations due to the manufacturing process and aging effects. In other terms, the design centering consists in assigning a set of values to the circuit parameters to maximize tolerances on parameters or to maximize the yield for an assumed statistical distribution [55-58].

4.1. Problem outline

Consider a n -dimensional point $e \in \mathfrak{R}^n$ that represents n independent system parameters (as circuit component values, MOSFET channel length and width, threshold voltage, etc.) and the set of design specifications, written in a standard manner [59]:

$$c(e, f_k) \equiv c_k(e) \leq 0 \quad k = 1, 2, \dots, K \quad (4)$$

where $\{f_k, k = 1, 2, \dots, K\}$ is a set of discretized continuous variables such as time, frequency, temperature, etc. Selection of the values of K and f_k is realized by the designer, so conservative overspecification makes some of the constraints (4) never active. The *acceptability region* A (or *feasible region*) in the space \mathfrak{R}^n is defined as the set of points e such that

$$A = \{e : c_k(e) \leq 0 \quad k = 1, 2, \dots, K\} \quad (5)$$

The points in A are named *feasible points*. The design center is the point $x \in A$ that maximizes the yield production [60]. **Fig. 11** shows the correlation between constraints, acceptability region and design center.

The problem of design centering is a task of the tolerance design and has been traditionally faced in two main ways: the geometrical approach and the statistical approach, even if several methods exist that hybridize these approaches [61, 62]. In the geometrical approach, the feasible region is approximated by a known geometrical body, such a polytope or an ellipsoid, then the center of this body, even if approximated, is taken to be the design center. This method has the disadvantages that:

- i) there are some limitations associated with the types of geometric bodies used to approximate the feasible region;
- ii) the design center could be highly dependent on the exact probability distributions of the variables and would change according to these distributions;
- iii) the hypothesis of convex feasible region is not always verified [63].

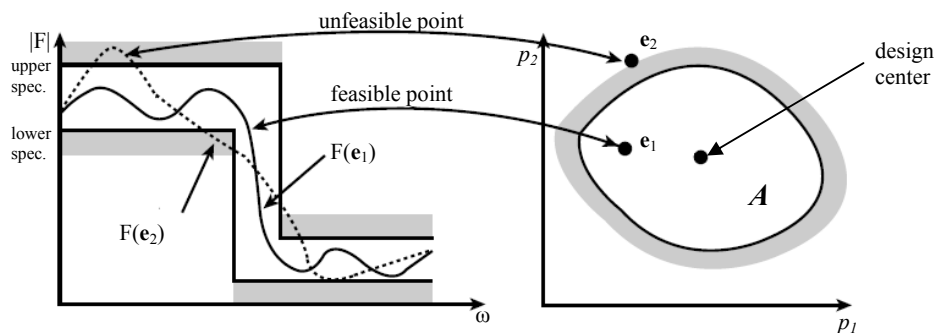


Figure 11: Illustration of the correlation between constraints and acceptability region and design center.

However, in the statistical approach, the overall performances of the solution space are estimated by simulating the circuit behavior for a sample of feasible points. The larger the sample, the more accurate the estimation will be. This method has the disadvantage of being computationally expensive when the targeted yield is high [64], but it gives a perfect accuracy if every point in the space is sampled.

4.2. The geometrical approaches

4.2.1. The simplicial approximation

This method is based on explicitly approximating the boundary, ∂R , of the feasible region, R , of an n -parameter design space by a polyhedron made up of n -dimensional simplices, and begins by determining any $m \geq n+1$ points $\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_m$ on the boundary ∂R [65]. After finding the polyhedron, it is possible to inscribe the largest hypersphere. The center of the largest hypersphere is the design center. **Fig. 12** shows the procedure for two dimensions and four iterations. This method of approximation is necessary since ∂R is generally known only in terms of nonlinear inequality constraints which express acceptable circuit performance in terms of voltage and current which depend implicitly upon the design parameters. An improvement of this method was developed in [66, 67] using quadratic approximation and was more suitable and successful than simplicial, but a unique quadratic approximation requires too many base points and thus too many circuits simulations.

4.2.2. The ellipsoidal method

This method is based on inscribing the largest possible ellipsoid into the constraint region. This is done in such a way that the end-points of the ellipsoid axes do not violate the constraints. It proceeds by generating a sequence of ellipsoids, each smaller than the last, until the procedure converges. The ellipsoid center provides the nominal design point. Similar to other methods, this procedure assumes that an initial feasible point is provided by the designer. **Fig. 13(a)** shows the ellipsoid method in a two dimensional problem.

4.2.3. Convexity-based approaches

In this case, the feasible region is initially approximated by a polytope, then it is possible to use the properties of polytope to inscribe the largest ellipsoid, or to formulate the design centering problem as a convex programming problem to use a convex programming algorithm to find the solution [63]. **Fig. 13(b)** shows the polytope approximation for a convexity-based method.

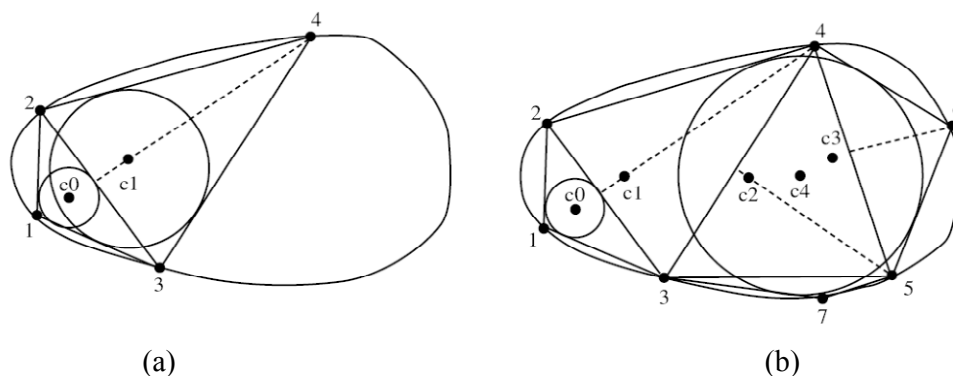


Figure 12: The simplicial approximation: (a) the initial polyhedron; (b) resulting polyhedron after four iterations.

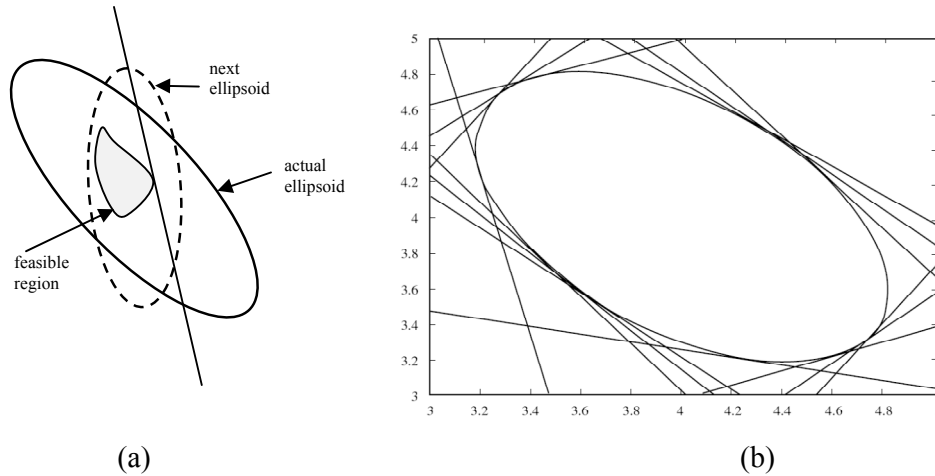


Figure 13: a) The ellipsoidal method. b) The convexity-based method.

4.3. The statistical approaches

The statistical approaches are based on the statistical methods of yield optimization. The objective of the yield optimization is to find a vector of designable parameters $\mathbf{p}=\mathbf{p}_{opt}$, such that $Y(\mathbf{p}_{opt})$ is maximized. In Fig. 14 is reported the interpretation of yield maximization for a discrete circuit with two parameters, where $p_{x,nom}$ are the nominal values of parameter design and $p_{x,opt}$ are the parameters values that maximize the yield.

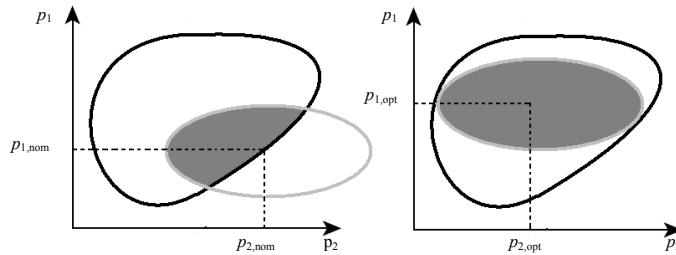


Figure 14: Graphical interpretation of yield maximization.

The statistical methods can be classified in two main groups: large-sample methods and small-sample methods. The large-sample methods calculate the expectation function from a large numbers of samples in the parameters space. On the other hand, the small-sample methods use just a few samples of the expectation function for any given point in the parameter space, but combining an averaging procedure to calculate the average of the expectation function or its gradient over a certain number of steps. The majority of yield optimization methods belongs to the large-sample groups.

4.4. The use of symbolic analysis in design centering

In any case, the design centering problem starts from the approximation of the acceptability region, otherwise we need to verify if a point \mathbf{p} in the parameter space is feasible or not, and this requires several circuit simulations by means of circuit simulator software. The use of symbolic analysis technique can give noteworthy advantages with respect to the numerical techniques in all the applications that require the repetition of a high number of simulations performed on the same circuit topology. In design centering problem they can be advantageously applied in both statistical and geometrical approaches [68-70].

Consider the problem of designing a linear, time-invariant circuit with k design parameters and h constraints. The circuit is represented by a network function $F(s, \mathbf{p})$ in the s domain, which, generally, is a transfer function. For the sake of simplicity, consider only constraints referred to the amplitude response of the circuit as follows:

$$|F(j\omega_i, \mathbf{p})| \leq K_i \text{ for } i=1 \dots h \quad (6)$$

where $\mathbf{p}=[p_1, p_2, \dots, p_k]$ is a point in the parameter space. A generic $F(j\omega, \mathbf{p})$ can be expressed as:

$$F(j\omega, \mathbf{p}) = \frac{a_n(\mathbf{p})(j\omega)^n + \dots + a_1(\mathbf{p})(j\omega) + a_0(\mathbf{p})}{b_m(\mathbf{p})(j\omega)^m + \dots + b_1(\mathbf{p})(j\omega) + b_0(\mathbf{p})} \quad (7)$$

where the coefficients of numerator and denominator can be considered as a sum of products (SOP) of the circuit parameters and the degree of this expression, with respect to a single parameter, is always equal to one. Then it is possible to write the expression (7) in bilinear form with respect to the q -th parameter as:

$$F(j\omega, p_q) = \frac{A(j\omega) + p_q B(j\omega)}{C(j\omega) + p_q D(j\omega)} \quad (8)$$

where the polynomials A, B, C and D depend only on the frequency, having fixed the numerical value for all the other parameters. Equation (8) can be expressed as a function of the real and imaginary parts of A, B, C and D as:

$$F(j\omega, p_q) = \frac{A_r(j\omega) + jA_i(j\omega) + p_q B_r(j\omega) + jp_q B_i(j\omega)}{C_r(j\omega) + jC_i(j\omega) + p_q D_r(j\omega) + jp_q D_i(j\omega)} \quad (9)$$

where the subscripts r and i of the polynomials A, B, C and D indicate the real and imaginary parts of the corresponding polynomials. At this point it is possible to solve the h inequalities in (6) with respect to the q -th parameter, when the other parameters have a fixed numerical value. Using the expression (9), the generic j -th inequality in (6) can be expressed in the following way:

$$|A_r + jA_i + p_q B_r + jp_q B_i| \leq K_j |C_r + jC_i + p_q D_r + jp_q D_i| \quad (10)$$

that is:

$$\sqrt{(A_r + p_q B_r)^2 + (A_i + p_q B_i)^2} \leq K_j \sqrt{(C_r + p_q D_r)^2 + (C_i + p_q D_i)^2} \quad (11)$$

where $A_r=A_r(j\omega_j)$, $A_i=A_i(j\omega_j)$, $B_r=B_r(j\omega_j)$, $B_i=B_i(j\omega_j)$, $C_r=C_r(j\omega_j)$, $C_i=C_i(j\omega_j)$, $D_r=D_r(j\omega_j)$, and $D_i=D_i(j\omega_j)$. Considering the square form of the expression (11), after easy calculations, the following inequality can be obtained:

$$a_j p_q^2 + b_j p_q + c_j \leq 0 \quad (12)$$

where:

$$\begin{aligned} a_j &= B_r^2 + B_i^2 - K_j^2 D_r^2 - K_j^2 D_i^2 \\ b_j &= 2(A_r B_r + A_i B_i - K_j^2 C_r D_r - K_j^2 C_i D_i) \\ c_j &= A_r^2 + A_i^2 - K_j^2 C_r^2 - K_j^2 C_i^2 \end{aligned} \quad (13)$$

In case of constrains defined as $|F(j\omega_j, \mathbf{p})| \geq K_j$, it is again easy to verify that:

$$a_j^* p_q^2 + b_j^* p_q + c_j^* \leq 0 \quad (14)$$

where $a_j^* = -a_j$, $b_j^* = -b_j$, $c_j^* = -c_j$. For each constraint, a range of values for the parameter p_q can be derived by the expression (12) (and/or the expression (14)), once the values of $A_r, A_i, B_r, B_i, C_r, C_i, D_r$ and D_i have been evaluated for each frequency of interest. This operation can be easily performed as follows, if the symbolic form of the polynomials A, B, C and D is available. Once the parameter unknown p_q has been chosen, and the frequency ω_j and all the other parameter values have been fixed, the numerator and the denominator of the expression (9) are evaluated for $p_q=1$ and for $p_q=0$, bringing to the following equations:

$$\begin{aligned}
 N_1 &= N(j\omega_j, p_q) \Big|_{p_q=1} = A_r + jA_i + B_r + jB_i \\
 D_1 &= D(j\omega_j, p_q) \Big|_{p_q=1} = C_r + jC_i + D_r + jD_i \\
 N_2 &= N(j\omega_j, p_q) \Big|_{p_q=0} = A_r + jA_i \\
 D_2 &= D(j\omega_j, p_q) \Big|_{p_q=0} = C_r + jC_i
 \end{aligned} \tag{15}$$

Now the numerical values of $A_r, A_i, B_r, B_i, C_r, C_i, D_r$ and D_i are the following:

$$\begin{aligned}
 A_r &= \text{Re}\{N_2\}; A_i = \text{Im}\{N_2\} \\
 C_r &= \text{Re}\{D_2\}; C_i = \text{Im}\{D_2\} \\
 B_r &= \text{Re}\{N_1 - N_2\}; B_i = \text{Im}\{N_1 - N_2\} \\
 D_r &= \text{Re}\{D_1 - D_2\}; D_i = \text{Im}\{D_1 - D_2\}
 \end{aligned} \tag{16}$$

This procedure, when applied to each parameter p_q and to each frequency ω_j , allows to determine the range of values satisfying the constraints for each circuit parameter at a time.

Starting from the previous theoretical considerations, the acceptability region can be represented through the determination of two-dimensional (2D) sections relevant to couples of parameters. The algorithm for determining a 2D section is explained in detail in [70]. In the algorithm a couple of parameters p_x and p_y is chosen and the shape of the acceptability region is investigated as a 2D cross-section relevant to these two parameters. Only p_x and p_y vary, while all the other parameters are fixed at their nominal value in \mathbf{p}' . For the parameter p_x a range of values relevant to a fixed percentage of its nominal value p_x' is considered. This range is divided in N_p samples. The percentage and N_p can be chosen by the user as a trade-off between accuracy of results and computational speed.

A pseudo 3D representation of the acceptability region can be simply obtained with the same procedure, performed through the superposition of the two-dimensional sections relevant to prefixed variations of a third parameter. Of course the procedure is applicable to the case of more than three variable parameters, but, obviously, it is not possible to plot the obtained region. For this reason the algorithm has been considered only for the cases of two and three dimensions. However, to extend the procedure to the case of a generic dimension $M > 3$, it is sufficient to modify the algorithm by inserting $M-1$ nested loop as in pseudo 3D representation.

It is important to note that, by exploiting symbolic simulation techniques, the circuit is simulated only once for determining the symbolic transfer function. During the procedure, only evaluations of the symbolic transfer function are required. For what concerns the algorithm complexity, in the 2D case, for each value of p_x , it is

necessary to solve a second degree equation and the number of values of p_x is $N_p \times h$. In the 3D case, the second degree equation has to be solved for $N_p^2 \times h$ times, while, for a generic MD (M -Dimensional) case, the second degree equation has to be solved for $N_p^{M-1} \times h$ times. Finally, it is worth to point out that here only the constraints referred to the amplitude response have been considered, because this kind of constraint is the most commonly used. Of course, other kinds of constraints can be considered, as the phase or the delay response, and the extension to this kind of responses is straightforward.

4.5. Application examples

The procedure described in the previous section has been implemented in the programs DESCEN (DESIGN CENTERING) [68] and SAR (Symbolic Acceptability Region) [69]. The programs acquire the symbolic transfer function from the program SAPWIN [2, 3], starting from a schematic entry of the circuit under consideration.

DESCEN allows to use both the geometrical and statistical approaches for searching the design center. Furthermore it allows to perform a yield estimation by using Monte Carlo analysis. The statistical approach consists in the implementation of the well-known center of gravity method (Gravity method), using the symbolic approach in Monte Carlo analysis. The verification of the constraints for each set of parameter values is performed by replacing numerical values in the symbolic transfer function without simulating the circuit for each set of parameter values.

As first example we consider the active tunable filter shown in **Fig.15(a)**. Using the constraints as reported in **Fig.15(b)** and the initial feasible point in [62], we obtain the results reported in **Table 1**, for 1% and 1.5% tolerances. For example, considering ideal the operational amplifier, the final values for R_1 , R_3 , C_1 and C_2 , with 1.5% component tolerance, are [12527.8 Ω , 184.34 Ω , 0.72727 μF , 0.73035 μF] with the Gravity method and [12489.8 Ω , 181.2 Ω , 0.7321 μF , 0.73621 μF] with the Volume method. In **Table 1** are reported, for the sake of comparison, also the results in [62] relevant to the Largest Hessian Ellipsoid (LHE) method and the Convex Programming (CP) method.

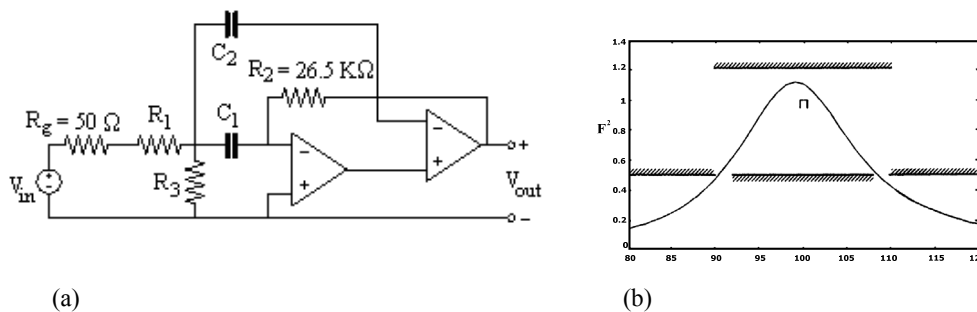


Figure 15: (a) Active tunable filter. (b) Required specifications.

Table 1: Yield Estimation for the Circuit in Fig. 15.

Component Tolerances (%)	Yield (%)			
	LHE	CP	Gravity (time)	Volume (time)
1	68.0	66.8	67.4 (107.13 s)	67.8 (6.32 s)
1.5	46.6	45.2	45.2 (112.06 s)	46.8 (8.40 s)

The second program, SAR, is able to determine the acceptability region of analog linear circuits. From the File menu of the program SAR, we can load the symbolic

form of the transfer function from the .fdt file generated by SAPWIN. In the constraint window we can insert the required specifications of the circuit under analysis. With the button Graf we obtain a first view of the acceptability region related to two circuit parameters, as shown, for example, in **Fig. 16(a)**. Now, we can choose the parameters and the constraints which are used to plot the acceptability region, using the Graf button as refresh.

In case of a circuit with three or more parameters it is possible to plot also the 3D shape of the acceptability region, by selecting the wanted parameters. In **Fig. 16(b)** an example of 3D plot is reported. The same circuit of previous example, shown in **Fig. 15(a)** is considered as example. The transfer function is given by:

$$F^2 = \left| \frac{V_2}{V_g} \right|^2 \quad (17)$$

The required specifications for the filter are shown in **Fig. 15(b)**. The response of the filter is evaluated at the nominal parameter values: $\{R_g=50\Omega, R_1=12.46 \text{ k}\Omega, R_2=26.5\Omega, R_3=184.3998\Omega, C_1=C_2=0.72855 \mu\text{F}\}$. Starting from the nominal values and using the algorithm in [70], it is possible to obtain a 2D section of the acceptability region. In **Fig. 16(a)**, a section of the acceptability region, relevant to the parameters C_1 and C_2 , is shown. Generally, as also in the case of **Fig.16**, the sections are obtained in a very short time on today's computers. The shaded area indicates the 2D section of the acceptability region relevant to the couple of parameters C_1 and C_2 . The program allows also to represent the contribution of a single constraint to the feasible region. In **Fig. 16(b)** an example of 3D plot is reported.

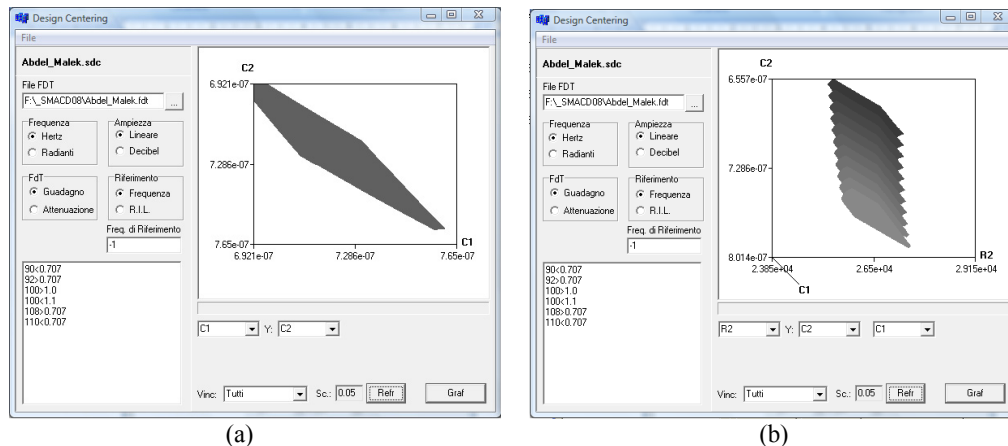


Figure 16 : Screenshots of SAR program: (a) 2D; (b) 3D.

5. Symbolic Analysis of PWM dc-dc Converters Considering True-Rms Parasitic Resistances

Pulse-Width Modulated (PWM) dc-dc power converters are circuits based on a controlled switch and a diode cyclically switching and driving the entire converter circuit through several topological configurations composed of linear reactive and resistive components, connected to a dc voltage source. There are two configurations for Continuous Conduction Mode (CCM) and three configurations for Discontinuous Conduction Mode (DCM). Often these circuit are used with a feedback circuit to regulate the output voltage. The design of this circuit is not trivial because PWM

converters actually are time variant circuits, and their s-domain representations require some model to be utilized.

The purpose of this section is to present a computer analysis of dc-dc PWM converters based on a small-signal model where parasitic resistances of the converter components are considered. By using a computer aided symbolic analysis, the influence of each single parasitic resistance on the converter operation is easily studied and, therefore, an accurate parametric analysis of the converter is performed.

A computer aided analysis allows for an automatic determination of the number of poles and zeros of the system and their values, the root loci and an automatic derivation of the four transfer functions describing the converter frequency domain behaviors, and plotting of related Bode plots.

5.1. The assumptions of the averaged model of switching cells

The model of the switching cell is developed under the following assumptions:

1. The transistor output capacitance and the diode capacitance are neglected, therefore switching losses are neglected;
2. The switch in the on state is modeled by a linear resistance r_{DS} and in the off state by an infinite resistance;
3. The diode in the on state is modeled by a linear battery V_F and a linear forward resistance r_F . The diode in the off state is modeled by an infinite resistance;
4. Passive components are considered linear, time-invariant, and frequency independent;
5. The output capacitance of the input voltage source is zero for both dc and ac components.

5.2. Averaged model and linearization of the CCM switching cell equivalent circuit

Fig. 17(a) shows the “switching cell” comprised of a controlled switch S and a diode D_S and its connection to the inductor L . This sub-circuit is the “core” part of a dc-dc switching converter. As shown in **Fig. 17(b)**, the combination of the controlled switch S and diode D_S acts as a device diverting the inductor current i_L through the switch when ON and through the diode when S is OFF.

Several averaged circuit models of the switching cells can be derived [71]. If it is assumed that these quantities vary at low frequencies, *i.e.*, below $f_s/2$, [72], the dc and ac components of the diode current and switch-to-inductor voltage drop are:

$$I_D = (1-D)I_L \quad (18)$$

$$V_{SL} = (1-D)V_{SD} \quad (19)$$

$$i_d \approx (1-D)i_L - I_L d \quad (20)$$

and

$$v_{sl} \approx (1-D)v_{sd} - V_{SD}d \quad (21)$$

respectively. The equivalent circuit corresponding to expressions (18)-(21) is the dc and the small-signal ac model of the switching sub-circuits. The derived ac equivalent circuit and that used as sub-circuits in computer simulation are shown in **Fig. 19(a)** and **Fig. 19(b)**, respectively.

The derived equivalent circuit is the small-signal ac model of the ideal switching cell (AC-ISC) shown in **Fig. 17(a)** which is also the sub-circuit utilized for the computer analysis of dc-dc and it is constituted by:

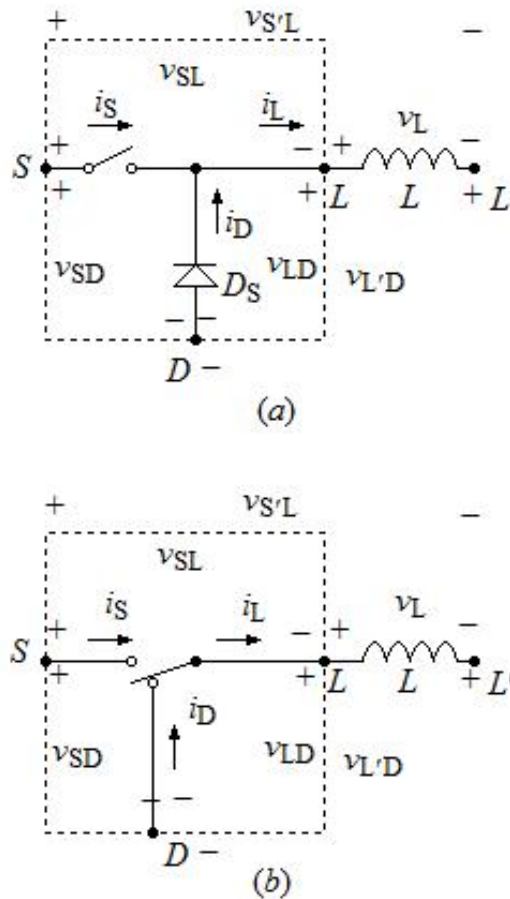


Figure 17: Switching cell of a dc-dc PWM converter: (a) switching cell composed of the diode and controlled switch; (b) switching cell composed of ideal switches.

- voltage-controlled voltage source $E1D$ - voltage source implementing $v'_{sl} = (1-D)v_{sd}$ with its positive and negative control nodes connected to ISC nodes S , node 1 in **Fig. 19(b)** and D , node 3 in **Fig. 19(b)**, respectively;

- voltage-controlled voltage source ESD - voltage source implementing $v''_{sl} = -V_{SD}d$. The negative control node is ground connected and the positive one is the external accessible node 4, which is the node where an independent voltage source is connected;

- current-controlled current source $F1D$, - source implementing $i'_d = (1-D)i_L$. Actually, the ac component of the inductor current flows through this source control branch (node 2 of the circuit shown in **Fig. 19(b)**);

- voltage-controlled current source $H1L$ - source implementing $i''_d = -I_Ld$. The control nodes are the same utilized for voltage source ESD .

The expressions for I_L and V_{SD} are given in **Table 2** for most common PWM dc-dc converters.

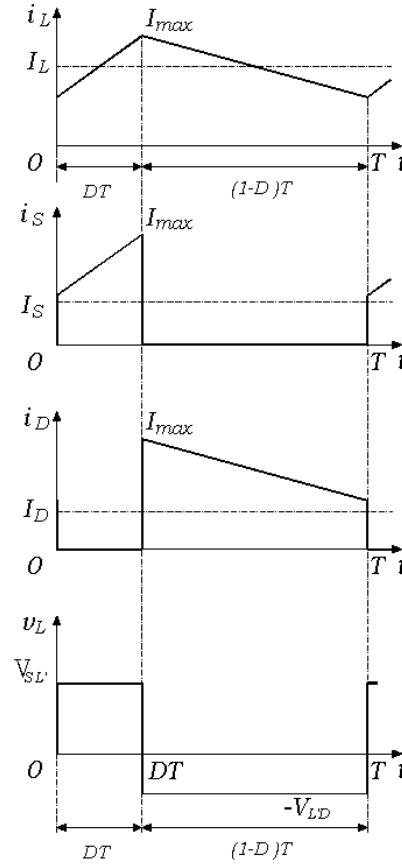


Figure 18: Waveforms of inductor, switch, and diode currents and inductor voltage in a dc-dc PWM converter operated in CCM.

Table 2 : Parameters of Controlled Sources for dc-dc Converter ac-model.

Converter Type	I_L	V_{SD}
Buck	I_O	$V_i - [r_{DSave} D - r_{Fave} (1-D)] I_L + V_F = V_i - [r_{DSave} D - r_{Fave} (1-D)] I_O + V_F$
Boost	$\frac{I_O}{1-D}$	$-V_o + [r_{DSave} D - r_{Fave} (1-D)] I_L - V_F = -V_o + \left[r_{DSave} \frac{D}{1-D} - r_{Fave} \right] I_O - V_F$
Buck-Boost	$\frac{I_O}{1-D}$	$V_i + V_o - [r_{DSave} D - r_{Fave} (1-D)] I_L - V_F = V_i - V_o - \left[r_{DSave} \frac{D}{1-D} - r_{Fave} \right] I_O - V_F$
Flyback	$\frac{I_O}{(1-D)}$	$\frac{V_L + V_o}{n} - \left[\frac{r_{DSave} + r_{T1ave}}{n^2} \frac{D}{1-D} - (r_{Fave} + r_{T2ave}) \right] I_O - V_F$

5.3. Averaging parasitic resistances of the CCM switching cell

From plots shown in **Fig. 18**, the expression of the rms values of the inductor, switch, and diode currents are derived. The expression of the power losses in these component equivalent resistances are $P_S = r_{DS} I_{Srms}^2$, $P_L = r_L I_{Lrms}^2$ and $P_D = r_F I_{Drms}^2$. If these losses are calculated in the averaged circuit of **Fig. 19**, the expression of these losses as functions of the averaged resistances are $P_S = r_{DSave} I_S^2$, $P_L = r_{Lave} I_L^2$ and $P_D = r_{Fave} I_D^2$.

Therefore, the averaged resistance are expressed as:

$$r_{Lave} = r_L \left[1 + \frac{V_O^2(1-D)^2}{12L^2 f^2 I_L^2} \right] \tag{22}$$

$$r_{DSave} = \frac{r_{DS}}{D} \left[1 + \frac{V_O^2(1-D)^2}{12L^2 f^2 I_L^2} \right] \tag{23}$$

and

$$r_{Dave} = \frac{r_F}{1-D} \left[1 + \frac{V_O^2(1-D)^2}{12L^2 f^2 I_L^2} \right] \tag{24}$$

When the converter is operated in CCM with $\Delta I \ll I_L$, that is $V_O(1-D)/\sqrt{12LfI_L} \ll 1$, (22), (23), and (24) simplify as $r_{Lave} \approx r_L$, $r_{DSave} \approx r_{DS}/D$ and $r_{Dave} \approx r_F/(1-D)$.

The equivalent resistances are considered in a Real Switching Cell (RSC) model where the averaged resistances are series connected to related components, that is r_{Lave} is series connected to the inductor, r_{DSave} series connected to the switch, and r_{Dave} series connected to the diode, as shown in Fig. 20.

Note that this sub-circuit remains the same for all the three more common non-insulated dc-dc converter topologies (buck, boost, and buck-boost).

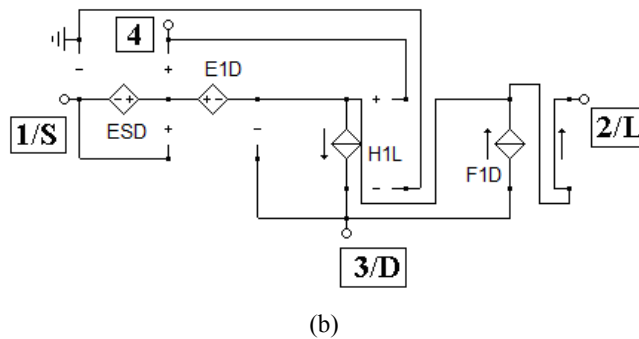
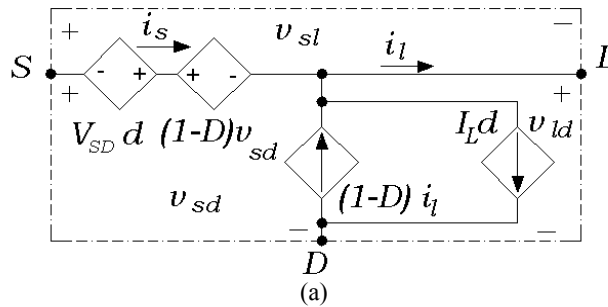


Figure 19: Equivalent circuit for the ac model of the switching cell of a dc-dc PWM converter: (a) equivalent circuit; (b) sub-circuit utilized in computer simulations.

Only parasitic component values which are external to the ISC are different between equivalent circuit of insulated converter and those related to their corresponding non-insulated circuits (*i.e.*, forward and buck converter). Therefore, the switching cell sub-circuit can also be usefully utilized for flyback and all the buck-derived dc-dc converters.

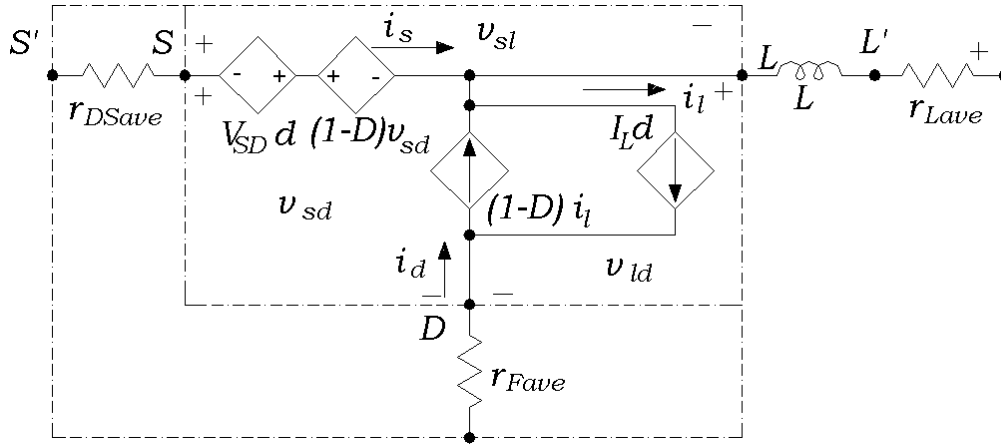


Figure 20: AC model of the Real Switching Cell (RSC).

Actually, the switching-cell sub-circuits yields to a unified approach to the frequency domain analysis of the dc-dc PWM converter circuits.

5.4. Averaged model and linearization of the DCM switching cell equivalent circuit

Detailed derivation of the expressions given in this paragraph are in [73]. By considering both the dc and small-signal ac components of switch current $i_s = I_S + i_s$, duty cycle $d_T = D + d$, and switch voltage $v_{SL} = V_{SL} + v_{sl}$, we have:

$$I_S = \frac{D^2 V_{SL}}{2L f_s}$$

and

$$i_s \approx \frac{D V_{SL}}{L f_s} d + \frac{D^2}{2L f_s} v_{sl} = k_i d + \frac{v_{sl}}{r_i}$$

where

$$k_i = \frac{D V_{SL}}{L f_s} \quad \text{and} \quad r_i = \frac{2L f_s}{D^2}$$

We also have:

$$i_d \approx \frac{D V_{SL}^2}{f_s L V_{LD}} d + \frac{D^2 V_{SL}}{f_s L V_{LD}} v_{sl} - \frac{D^2 V_{SL}^2}{2 f_s L V_{LD}^2} v_{ld} = k_o d + g_m v_{sl} - \frac{v_{ld}}{r_o}$$

where

$$k_o = \frac{D V_{SL}^2}{f_s L V_{LD}}, \quad g_m = \frac{D^2 V_{SL}}{f_s L V_{LD}} \quad \text{and} \quad r_o = \frac{2 f_s L V_{LD}^2}{D^2 V_{SL}^2}.$$

The small-signal ac model of the switching sub-circuit is shown in **Fig. 21(b)**.

5.5. Averaging parasitic resistances of the DCM switching cell

From plots shown in **Fig. 22**, the expression of the rms values of the inductor, switch, and diode currents are derived so that the expression for the inductor equivalent series resistance is:

$$r = r_{Lav} + r_{DSav} + r_{Fav} = \frac{4}{3} \frac{1}{D + D_1} \left(r_L + \frac{D}{D + D_1} r_{DS} + \frac{D_1}{D + D_1} r_F \right) = \frac{4}{3} \frac{1}{D \left(\frac{V_{SL}}{V_{LD}} + 1 \right)} \left[r_L + \frac{r_{DS}}{\left(\frac{V_{SL}}{V_{LD}} + 1 \right)} + \frac{r_F}{\left(\frac{V_{LD}}{V_{SL}} + 1 \right)} \right]$$

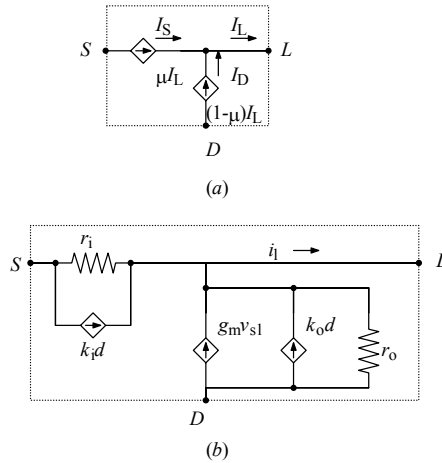


Figure 21 : Model of the switching cell of a dc-dc PWM converter: (a) dc model; (b) small-signal model.

The power loss due to the diode threshold voltage V_F is:

$$P_{DV} = V_F I_D = V_F \frac{D_1}{D + D_1} I_L$$

which leads to the equivalent averaged voltage source connected in series with inductor L :

$$V_{Fav} = \frac{D_1}{D + D_1} V_F = \frac{V_F}{\frac{D}{D_1} + 1} = \frac{V_F}{\frac{V_{LD}}{V_{SL}} + 1} = (1 - \mu) V_F$$

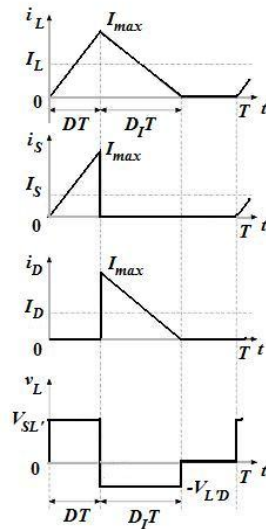


Figure 22 : Waveforms of inductor, switch, and diode currents and inductor voltage in a dc-dc PWM converter operated in DCM.

5.6. Applications examples

Models shown in previous section are useful for the frequency domain analysis of PWM dc-dc converters. Detailed information on frequency domain behaviors of these converters are given by four functions as follows.

The control-to-output transfer function is:

$$T_p(s) = \left. \frac{v_o(s)}{d(s)} \right|_{v_i=0}$$

The input-to-output voltage transfer function is:

$$M_v(s) = \left. \frac{v_o(s)}{v_i(s)} \right|_{d=0}$$

The output impedance is:

$$Z_o(s) = \left. \frac{v_o(s)}{i_o(s)} \right|_{d=0, v_i=0}$$

The input impedance is:

$$Z_i(s) = \left. \frac{v_i(s)}{i_i(s)} \right|_{d=0}$$

These functions highly depend on parasitic components. When these components are considered, the expression of these functions become cumbersome as shown in the following example. Computer programs such as SAPWIN which can automatically derive these expressions become useful when parasitics are considered and/or cannot be neglected.

5.6.1. PWM boost converter DCM operated

A circuit of a PWM boost converter is shown in Fig. 23(a, b and c) show the converter equivalent dc and ac circuits, respectively, where r_C is the Equivalent Series Resistance (ESR) of the filter capacitor.

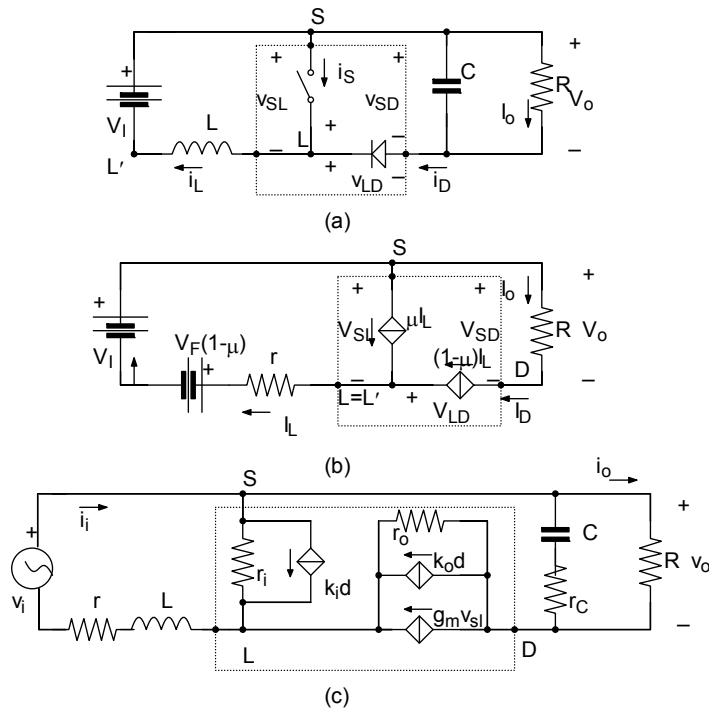


Figure 23: DC-DC PWM boost converter: (a) converter circuit; (b) dc equivalent circuit; (c) small-signal model.

Small-signal model

The small-signal model parameters for the boost converter are :

$$\begin{aligned}
 k_i &= \frac{DV_{SL}}{Lf_s} = \frac{D}{Lf_s}(V_I - rI_L) \\
 r_i &= \frac{1}{g_i} = \frac{2Lf_s}{D^2} \\
 g_m &= \frac{D^2V_{SL}}{f_sLV_{LD}} = \frac{D^2(V_I - rI_L)}{f_sL[V_O - V_I + rI_L - V_F(1-\mu)]} = \frac{2I_D}{V_{SL}} = \frac{2I_O}{V_I - rI_L} \\
 k_o &= \frac{DV_{SL}^2}{f_sLV_{LD}} = \frac{D(V_I - rI_L)^2}{f_sL[V_O - V_I + rI_L - V_F(1-\mu)]} = \frac{2I_D}{D} = \frac{2I_O}{D} \\
 r_o &= \frac{1}{g_o} = \frac{2f_sLV_{LD}^2}{D^2V_{SL}^2} = \frac{2f_sL[V_O - V_I + rI_L - V_F(1-\mu)]^2}{D^2(V_I - rI_L)^2} = \frac{V_{LD}}{I_D} = \frac{V_O - V_I + rI_L - V_F(1-\mu)}{I_O}
 \end{aligned}$$

and

$$\begin{aligned}
 r &= \frac{4}{3(D+D_1)} \left(r_L + \frac{Dr_{DS} + D_1r_F}{D+D_1} \right) = \frac{4}{3D\left(\frac{V_{SL}}{V_{LD}} + 1\right)} \left(r_L + \frac{r_{DS} + \frac{V_{SL}}{V_{LD}}r_F}{\frac{V_{SL}}{V_{LD}} + 1} \right) \\
 &= \frac{4}{3D\left[\frac{V_I - rI_L - V_F(1-\mu)}{V_O - V_I + rI_L + V_F(1-\mu)} + 1\right]} \left[r_L + \frac{r_{DS} + \frac{V_I - rI_L - V_F(1-\mu)}{V_O - V_I + rI_L + V_F(1-\mu)}r_F}{\frac{V_I - rI_L - V_F(1-\mu)}{V_O - V_I + rI_L + V_F(1-\mu)} + 1} \right]
 \end{aligned}$$

Small-signal transfer functions

The control-to-output transfer function for the boost converter is:

$$T_p(s) = \frac{v_o(s)}{d(s)} \Big|_{v_i=0} = T_{px} \frac{(s + \omega_{z1})(s - \omega_{z2})}{s^2 + 2\xi\omega_0s + \omega_0^2} = T_{px} \frac{(s - z_1)(s - z_2)}{(s - p_1)(s - p_2)}$$

where

$$T_{px} = \frac{r_c R [k_o r_o - k_i r_i (1 + g_m r_o)]}{r_c R + (r_i + r_o + g_m r_i r_o)(R + r_c)}$$

$$z_1 = -\omega_{z1} = -\frac{1}{r_c C}$$

$$z_2 = \omega_{z2} = \frac{k_o r_o (r + r_i) - k_i r_i (1 + g_m r_o)}{L [k_i r_i (1 + g_m r_o) - k_o r_o]}$$

$$\omega_0 = \sqrt{\frac{R(r + r_i) + r(r_i + r_o + g_m r_i r_o) + r_i r_o}{LC [Rr_c + (R + r_c)(r_i + r_o + g_m r_i r_o)]}}$$

$$\xi = \frac{L(R + r_i + r_o + g_m r_i r_o) + C \{ Rr_c (r + r_i) + (R + r_c) [r_i r_o + r(r_i + r_o + g_m r_i r_o)] \}}{2\sqrt{LC [Rr_c + (R + r_c)(r_i + r_o + g_m r_i r_o)]} [R(r + r_i) + r(r_i + r_o + g_m r_i r_o) + r_i r_o]}$$

and

$$p_1, p_2 = -\omega_0 \xi \left(1 \pm \sqrt{1 - 1/\xi^2} \right)$$

The transfer function T_p may have a zero in the RHP. Thus, the boost converter may become a non-minimal system, in which case it is more difficult to achieve stable operation and good dynamic performance.

The input-to-output voltage transfer function is:

$$M_v(s) = \left. \frac{v_o(s)}{v_i(s)} \right|_{d=0} = M_{vx} \frac{s - z_1}{s^2 + 2\xi\omega_0 s + \omega_0^2}$$

where

$$M_{vx} = \frac{Rr_c r_i (1 + g_m r_o)}{L [Rr_c + (R + r_c)(r_i + r_o + g_m r_i r_o)]}$$

The output impedance is given by:

$$Z_o(s) = \left. \frac{v_o(s)}{i_o(s)} \right|_{\substack{d=0 \\ v_i=0}} = Z_{ox} \frac{(s + \omega_{z1})(s + \omega_{z3})}{s^2 + 2\xi\omega_0 s + \omega_0^2} = Z_{ox} \frac{(s - z_1)(s - z_3)}{(s - p_1)(s - p_2)}$$

where

$$Z_{ox} = \frac{Rr_c (r_i + r_o + g_m r_i r_o)}{Rr_c + (R + r_c)(r_i + r_o + g_m r_i r_o)}$$

$$z_3 = -\omega_{z3} = -\frac{r_i r_o + r(r_i + r_o + g_m r_i r_o)}{L(r_i + r_o + g_m r_i r_o)}$$

The input impedance is given by:

$$Z_i(s) = \left. \frac{v_i(s)}{i_i(s)} \right|_{d=0} = Z_{ix} \frac{s^2 + 2\xi\omega_0 s + \omega_0^2}{s + \omega_{pzi}}$$

where

$$Z_{ix} = L$$

and

$$\omega_{pzi} = \frac{R + r_i + r_o + g_m r_i r_o}{C [Rr_c + (r_i + r_o + g_m r_i r_o)(R + r_c)]}$$

At $s = 0$:

$$Z_{io} = Z_i(0) = Z_{ix} \frac{\omega_0^2}{\omega_{pzi}} = \frac{R(r + r_i) + r(r_i + r_o + g_m r_i r_o) + r_i r_o}{R + r_i + r_o + g_m r_i r_o}$$

5.6.2. CCM operated converter feedback loop design examples

A PWM buck dc-dc with a switching frequency $f_s=100$ kHz, a dc input voltage $V_I=12.5$ V, a 5 W output power and a nominal output voltage $V_O = 5$ V is considered as example. The converter nominal duty cycle is $D=0.4$ and the load resistance $R=5$ Ω . The converter inductance is $L=150$ μ H and the output capacitance is $C=50$ μ F. The parasitic components of the converter equivalent circuit are $r_L=100$ m Ω , $r_{DS}=100$ m Ω , $r_F=30$ m Ω , $r_C=100$ m Ω .

Fig. 24 shows the equivalent circuits of a PWM buck converter based on the RSC switching cell model. **Fig. 25** shows the block diagram of a dc-dc PWM converter with $K_{FM} =$ PWM controller transfer function:

$$K_{FM} \equiv \frac{d(s)}{v_c(s)} = \frac{1}{V_{CM}}$$

where V_{CM} is the maximum value of the voltage saw-tooth waveform generated by the PWM controller; $A(s)$ is the compensating network transfer function and depends on the feedback loop.

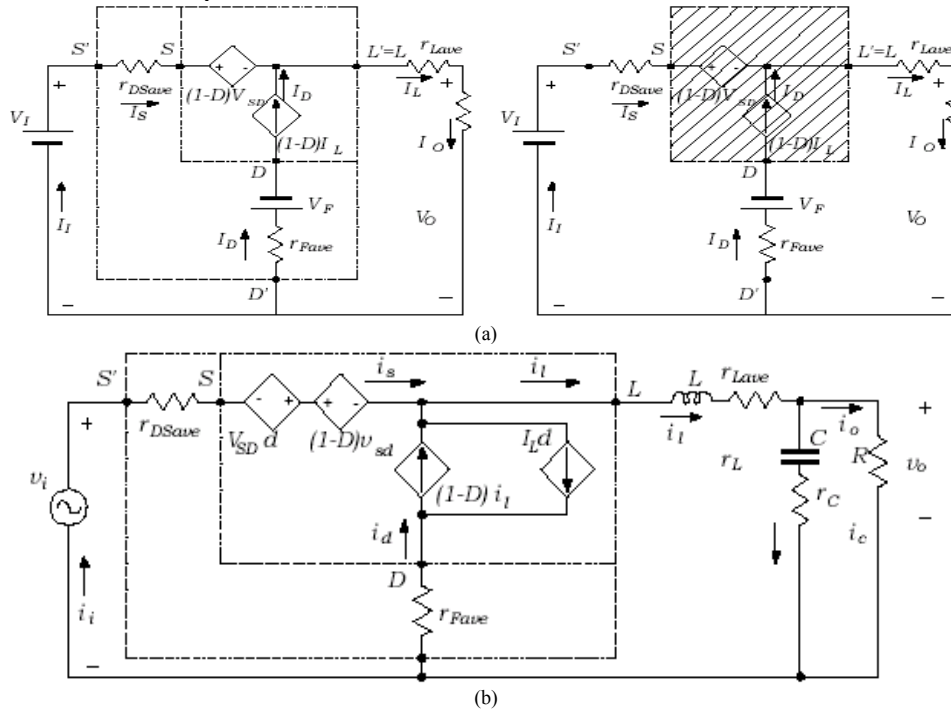


Figure 24: Averaged-circuit models of the PWM buck converter: (a) dc equivalent circuit; (b) small signal circuit.

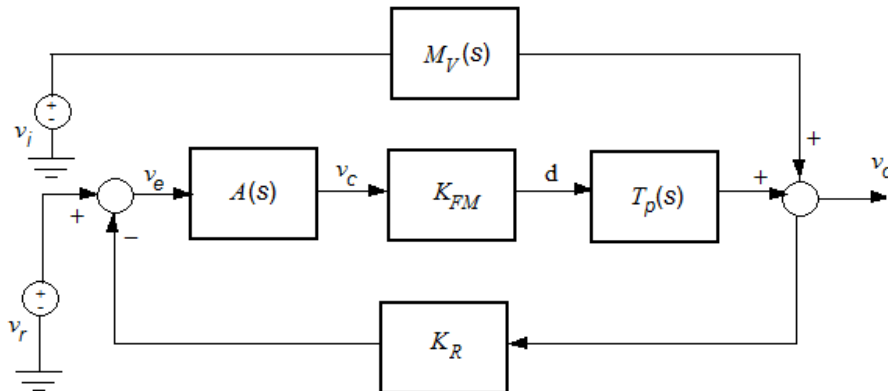
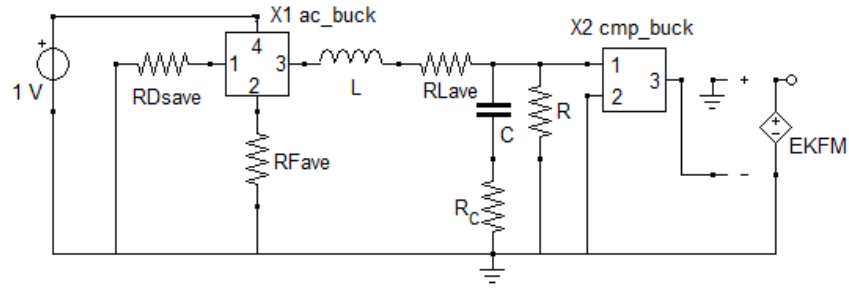


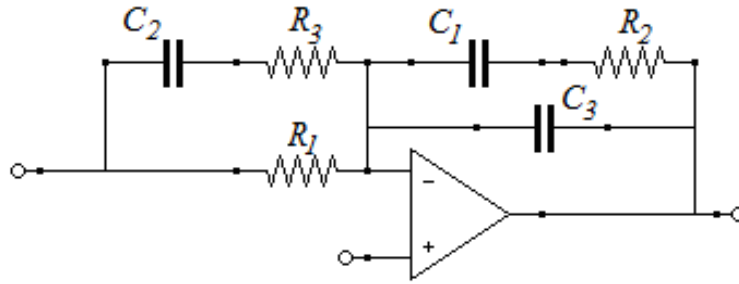
Figure 25: Block diagram of a PWM linearized dc-dc converter.

The output voltage v_o is measured, using a sensor with gain K_R . The sensor circuit is a voltage divider, comprised of precision resistor. The sensor output $K_R v_o$ is compared with a reference voltage v_r . The objective is to make $K_R v_o$ equal to v_r , so that v_o accurately follows v_r regardless of disturbances or component variations in the compensator, PWM, gate driver or converter power stage. The error signal v_e obtained by the difference between $K_R v_o$ and v_r is usually nonzero but nonetheless small.

The open loop equivalent circuit simulated by SAPWIN is shown in Fig. 26(a, b) shows the feedback network resulting in $A(s)$.



(a)



(b)

Figure 26 : (a) Open loop equivalent circuit. (b) Feedback network sub-circuit X2 cmp_buck.

Typically, the procedure for controller design refers to the “loop gain”:

$$T_{ol}(s) = K_R A(s) T_p(s) / V_M$$

The first zero of the compensating network is placed between 0 to $\frac{1}{2}$ of converter resonance frequency $f_{CC} = 1/2 \pi (LC)^{1/2}$. Therefore we have:

$$f_{z1} = 700 \text{ Hz}.$$

The second zero frequency is chosen between f_{z1} and f_{CC} :

$$f_{z2} = 1200 \text{ Hz}.$$

The first pole is placed at zero frequency; the second one is placed at the cross-frequency of the filter capacitor:

$$f_{p2} = f_{ESR} = \frac{1}{2\pi C R_c} = 31830 \text{ Hz}$$

The third pole frequency is placed at $f_s/2$ to reduce the output voltage ripple due to the switching:

$$f_{p3} = \frac{f_s}{2} = 50000 \text{ Hz}$$

$$f_{p3} = \frac{f_s}{2} = 50000 \text{ Hz}$$

The open loop bandwidth is chosen inside the interval $1/10 f_s$ and $1/5 f_s$ trying to get the highest possible value for $T_p(s)$ cross-over frequency. In this example it has been chosen:

$$f_{CO} = 20000 \text{ Hz}.$$

From these frequency values the components of the circuit shown in **Fig. 26(b)** are calculated as follows: $R_2 = 20 \text{ k}\Omega$, $C_1 = 1/2\pi R_2 f_{z1} = 11 \text{ nF}$, $C_2 = 1/2\pi R_2 f_{CO} = 400 \text{ pF}$, $C_3 = 1/2\pi R_2 f_{p3} = 159 \text{ pF}$, $R_1 = 1/2\pi C_2 f_{z2} = 120 \text{ k}\Omega$, $R_3 = 1/2\pi C_2 f_{p2} = 4.55 \text{ k}\Omega$.

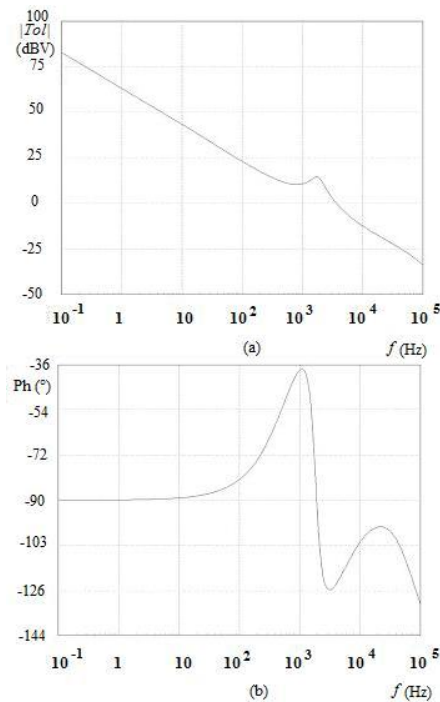


Figure 27 : Loop gain transfer function Bode plots T_{ol} : (a) magnitude; (b) phase.

As shown in **Fig. 27**, these component values yields to a phase margin $\phi_m = 57^\circ$ and a bandwidth $B_w = 3900$ Hz .

References

- [1] A. Liberatore, A. Luchetta, S. Manetti and M.C. Piccirilli, "A new symbolic program package for the interactive design of analog circuits", in *IEEE International Symposium on Circuits and Systems*, 1995, pp. 2209-2212.
- [2] A. Luchetta, S. Manetti and A. Reatti, "SAPWIN - A Symbolic Simulator as a Support in Electrical Engineering Education", *IEEE Transactions on Education*, vol. 44, pp. 213–213 and enclosed CD-ROM issue, May 2001.
- [3] L.P. Huelsman, "SAPWIN, Symbolic Analysis Program for Windows - PC Programs for Engineers", *IEEE Circuits and Devices Magazine*, vol. 6, pp. 4-6, March 1996.
- [4] L.P. Huelsman, "Symbolic analysis – a tool for teaching undergraduate circuit theory", *IEEE Transaction on Education*, vol. 39, pp. 243-250, May 1996.
- [5] J.B. Grimbleby, "Algorithm for finding the common spanning trees of two graphs", *Electronics Letters*, vol. 17, pp. 470-471, June 1981.
- [6] B.S. Rodanski, "Modification of the two-graph method for symbolic analysis of circuits with non-admittance elements", in *International Conference of Signals and Electronic Systems*, 2002, pp. 249-254.
- [7] E. Cengelci, "Software makes transfer functions more manageable", *Power Electronics Technology Magazine* vol. 34, pp. 14-21, June 2008.
- [8] J. Vlach, K. Singhal, *Computer methods for circuit analysis and design*. New York: Van Nostrand Reinhold, 1994.
- [9] R. S. Berkowitz, "Conditions for network-element-value solvability", *IEEE Transactions on Circuit Theory*, vol. 9, pp. 24-29, 1962.
- [10] R. Saeks, "A measure of testability and its application to test point selection theory", in *the 20th Midwest Symposium on Circuits and Systems*, 1977, pp. 576-583.
- [11] N. Sen and R. Saeks, "Fault diagnosis for linear systems via multifrequency measurement", *IEEE Transactions on Circuits and Systems*, vol. 26, pp. 457-465, July 1979.

- [12] H. M. S. Chen and R. Saeks, "A search algorithm for the solution of multifrequency fault diagnosis equations", *IEEE Transactions on Circuits and Systems*, vol. 26, pp. 589-594, July 1979.
- [13] R. Saeks et al., "Fault analysis in electronic circuits and systems", Technological Report, Institute for Electronics Science, Texas Technological University, Lubbock, 1978.
- [14] G. Temes, "Efficient method of fault simulation", in *the 20th Midwest Symposium on Circuits and Systems*, 1977, pp. 191-194.
- [15] W. J. Dejka, "A review of measures of testability for analog systems", in *AUTOTESTCON*, 1977, pp. 115-122.
- [16] R. W. Priester and J. B. Clary, "New measures of testability and test complexity for linear analog failure analysis", *IEEE Transactions on Circuits and Systems*, vol. 28, pp. 1088-1092, November 1981.
- [17] J. W. Bandler and A. E. Salama, "Fault diagnosis of analog circuits", *Proceedings of the IEEE*, vol. 73, pp. 1279-1325, August 1985.
- [18] J. A. Starzyk and H. Dai, "Multifrequency measurements of testability in analog circuits", in *IEEE International Symposium on Circuits and Systems*, 1987.
- [19] G. N. Stenbakken and T. M. Souders, "Test point selection and testability measures via QR factorization of linear models", *IEEE Transactions on Instrumentation and Measurement*, vol. 36, pp. 406-410, February 1987.
- [20] G. N. Stebbakken, T. M. Souders and G. W. Stewart, "Ambiguity groups and testability", *IEEE Transactions on Instrumentation and Measurement*, vol. 38, pp. 941-947, October 1989.
- [21] G. Fedi, S. Manetti, M. C. Piccirilli and J. Starzyk, "Determination of an optimum set of testable components in the fault diagnosis of analog linear circuits", *IEEE Transactions on Circuits and Systems I*, vol. 46, pp. 779-787, July 1999.
- [22] G. Fedi, S. Manetti and M. C. Piccirilli, "Comments on Linear circuit fault diagnosis using neuromorphic analyzers", *IEEE Transactions on Circuits and Systems II*, vol. 46, pp. 483-485, April 1999.
- [23] G. Iuculano, A. Liberatore, S. Manetti and M. Marini, "Multifrequency measurement of testability with application to large linear analog systems", *IEEE Transactions on Circuits and Systems*, vol. 23, pp. 644-648, June 1986.
- [24] M. Catelani, G. Iuculano, A. Liberatore, S. Manetti and M. Marini, "Improvements to numerical testability evaluation", *IEEE Transactions on Instrumentation and Measurement*, vol. 36, pp. 902-907, December 1987.
- [25] R. Carmassi et al., "Analog network testability measurement: a symbolic formulation approach", *IEEE Transactions on Instrumentation and Measurement*, vol. 40, pp. 930-935, December 1991.
- [26] A. Liberatore, S. Manetti and M. C. Piccirilli, "A new efficient method for analog circuit testability measurement", in *IEEE Instrumentation and Measurements Technical Conference*, 1994, pp. 193-196.
- [27] M. Catelani et al., "A new symbolic approach for testability measurement of analog networks", in *Mediterranean Electrotechnical Conference*, 1996, pp. 517-520.
- [28] G. Fedi, A. Luchetta, S. Manetti and M. C. Piccirilli, "A new symbolic method for analog circuit testability evaluation", *IEEE Transactions on Instrumentation and Measurement*, vol. 47, pp. 554-565, April 1998.
- [29] S. Manetti and M. C. Piccirilli, "A singular-value decomposition approach for ambiguity group determination in analog circuits", *IEEE Transactions on Circuits and Systems I*, vol. 50, pp. 477-487, April 2003.
- [30] F. Grasso, S. Manetti and M. C. Piccirilli, "A program for ambiguity group determination in analog circuits using singular-value decomposition", in *European Conference on Circuit Theory and Design*, 2003, pp. 57-60.
- [31] A. Liberatore and S. Manetti, "SAPEC - A personal computer program for the symbolic analysis of electric circuits", in *IEEE International Symposium on Circuits and Systems*, 1988, pp. 897-900.
- [32] S. Manetti, "A new approach to automatic symbolic analysis of electric circuits", *IEE Proceedings Part G: Circuits, Devices and Systems*, vol. 138, pp. 22-28, February 1991.
- [33] A. Liberatore and S. Manetti, "Network sensitivity analysis via symbolic formulation", in *IEEE International Symposium on Circuits and Systems*, 1989, pp. 705-708.
- [34] G. Fedi, R. Giomi, A. Luchetta, S. Manetti and M. C. Piccirilli, "Symbolic algorithm for ambiguity group determination in analog fault diagnosis", in *European Conference on Circuit Theory and Design*, 1997, pp. 1286-1291.

- [35] G. Fedi, R. Giomi, A. Luchetta, S. Manetti and M. C. Piccirilli, "On the application of symbolic techniques to the multiple fault location in low testability analog circuits", *IEEE Transactions on Circuits and Systems II*, vol. 45, pp. 1383-1388, October 1998.
- [36] J. Starzyk *et al.*, "A software program for ambiguity group determination in low testability analog circuits", in *European Conference on Circuit Theory and Design*, 1999, pp.603-606.
- [37] J. Starzyk, J. Pang, S. Manetti, M. C. Piccirilli and G. Fedi, "Finding ambiguity groups in low testability analog circuits", *IEEE Transactions on Circuits and Systems I*, vol. 47, pp. 1125-1137, August 2000.
- [38] G. H. Golub, C. F. Van Loan, *Matrix computations*, Baltimore, Maryland, John Hopkins University Press, 1983.
- [39] R. Liu, *Testing and diagnosis of analog circuits and systems*, New York, Van Nostrand Reinhold, 1991.
- [40] J. L. Huertas, "Test and design for testability of analog and mixed-signal integrated circuits: theoretical basis and pragmatical approaches", in *European Conference on Circuit Theory and Design*, 1993, pp. 75-151.
- [41] G. Fedi, A. Liberatore, A. Luchetta, S. Manetti and M. C. Piccirilli, "A symbolic approach to the fault location in analog circuits", in *IEEE International Symposium on Circuits and Systems*, 1996, pp. 810-813.
- [42] M. Catelani *et al.*, "A new symbolic approach to the fault diagnosis of analog circuits", in *IEEE Instrumentation and Measurements Technical Conference*, 1996, pp. 1182-1185.
- [43] M. Catelani *et al.*, "A fully automated measurement system for the fault diagnosis of analog electronic circuits", in *XIV IMEKO World Congress*, 1997, pp. 52-57.
- [44] G. Fedi, A. Luchetta, S. Manetti and M. C. Piccirilli, "Multiple fault diagnosis of analog circuits using a new symbolic approach", in *International Workshop on Symbolic Methods and Applications in Circuit Design*, 2000, pp. 139-143.
- [45] F. Grasso, A. Luchetta, S. Manetti and M. C. Piccirilli, "Recent advances in symbolic techniques for analog fault diagnosis", in *International Workshop on Symbolic Methods and Applications in Circuit Design*, 2002, pp. 21-24.
- [46] F. Grasso, A. Luchetta, S. Manetti and M. C. Piccirilli, "Symbolic techniques for the selection of test frequencies in analog fault diagnosis", *Analog Integrated Circuits and Signal Processing*, vol. 40, pp. 205-213, September 2004.
- [47] F. Grasso, S. Manetti and M. C. Piccirilli, "An approach to analog fault diagnosis using genetic algorithms", in *Mediterranean Electrotechnical Conference*, 2004, pp. 111-114.
- [48] F. Grasso, A. Luchetta, S. Manetti and M.C. Piccirilli, "A method for the automatic selection of test frequencies in analog fault diagnosis", *IEEE Transactions on Instrumentation and Measurement*, vol. 56, pp. 2322-2329, December 2007.
- [49] S. Manetti, and M.C. Piccirilli, "Symbolic function approaches for analogue fault diagnosis," in *Test and diagnosis of analogue, mixed-signal and RF integrated circuits*, Yichuang Sun, Ed. London: IET-The Institution of Engineering and Technology, 2008, Ch. 2.
- [50] S. Manetti and M.C. Piccirilli, "Symbolic simulators for the fault diagnosis of nonlinear analog circuits", *Analog Integrated Circuits and Signal Processing*, vol. 3, pp. 59-72, January 1993.
- [51] G. Fedi, R. Giomi, S. Manetti and M. C. Piccirilli, "A symbolic approach for testability evaluation in fault diagnosis of nonlinear analog circuits", *IEEE International Symposium on Circuits and Systems*, 1998, pp. 9-12.
- [52] A. Luchetta, S. Manetti and M. C. Piccirilli, "Critical comparison among some analog fault diagnosis procedures based on symbolic techniques", in *Design, Automation and Test in Europe*, 2002, pp. 1105.
- [53] F. Grasso, A. Luchetta, S. Manetti and M. C. Piccirilli, "Symbolic techniques in parametric fault diagnosis of analog circuits", in *Baltic Electronics Conference*, 2002, pp. 271-274.
- [54] B. Cannas, A. Fanni, S. Manetti, A. Montisci and M. C. Piccirilli, "Neural network-based analog fault diagnosis using testability analysis", *Neural Computing and Applications*, vol. 13, pp. 288-298, December 2004.
- [55] R. Spence, R.S. Soin, *Tolerance design of integrated circuits*. Reading, MA: Addison-Wesley, 1988.
- [56] M.A. Styblinski, "Statistical design optimization," in *The Circuits and Filters Handbook*, Chen WK, Ed. Boca Raton, FL: CRC Press; 2003, Ch. 50.
- [57] H. E. Graeb, *Analog design centering and sizing*, Springer, 2007.
- [58] H.L. Abdel-Malek, A-K. S.O. Hassan, and M.H. Heaba, "A boundary Gradient Search Technique and Its Applications in Design Centring", *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, vol.18, pp. 1654 – 1661, November 1999.

- [59] J. W. Bandler and S. A. Chen, "Circuit optimization: the state of the art", *IEEE Transactions on Microwave Theory and Technique*, vol. 36, pp. 424-442, February 1988.
- [60] S.S. Sapatnekar, "Design by optimization," in *The Circuits and Filters Handbook*, Chen WK, Ed. Boca Raton, FL: CRC Press; 2003, Ch. 49.
- [61] P. Feldman and S.W. Director, "Accurate and efficient evaluation of circuit yield and yield gradients", in *International Conference on Computer-Aided Design*, 1991, pp. 120-123.
- [62] M.C. Bernardo et al., "Integrated circuit design optimization using a sequential strategy", *IEEE Transactions on Computer Aided Design*, vol.11, pp. 361-372, March 1992.
- [63] S.S. Sapatnekar, P.M. Vaidya, and S-M Kang, "Convexity-based algorithms for design centering", *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, vol.13, pp. 1536-1549, December 1994.
- [64] H.L. Abdel-Malek, A-K. S.O. Hassan, and M.H. Heaba, "The ellipsoidal technique for design centering and region approximation", *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, vol.10, pp. 1006-1014, August 1991.
- [65] S.W. Director, and G.D. Hachtel, "The simplicial approximation approach to desing centering", *IEEE Transactions on Circuits and Systems*, vol. 24, pp. 363-372, July 1977.
- [66] J. Bandler, and H. Abdel-Malek, "Optimal centering, tolerancing, and yield determination via updated approximations and cuts", *IEEE Transactions on Circuits and Systems*, vol. 25, pp. 853-871, October 1978.
- [67] D.E. Hocevar, M.R. Lightner, and T.N. Trick, "Monte carlo based yield maximization with quadratic model", in *IEEE International Symposium on Circuits and Systems*, 1983, pp. 550-553.
- [68] F. Grasso, S. Manetti, and M.C. Piccirilli, "A symbolic approach to design centering of analog circuits", *Microelectronics Reliability*, vol. 47, pp. 1288-1295, August 2007.
- [69] F. Grasso, S. Manetti, and M.C. Piccirilli, "SAR: a symbolic program for accteptability region representation in analog circuit design", in *Xth International Workshop on Symbolic and Numerical Methods, Modeling and Applications to Circuit Design*, 2008, pp 142-148.
- [70] F. Grasso, S. Manetti, and M.C. Piccirilli, "A method for acceptability region representation in analogue linear networks", *International Journal on Circuit Theory and Applications*, vol.37, pp. 1051-1061, December 2009.
- [71] M. K. Kazimierzuk and D. Czarkowski, "Application of the principle of energy conservation modeling the PWM converters", in *2nd IEEE Conference on Control Applications*, 1993, pp. 291-296.
- [72] J. Sun, D. M. Mitchell, M.F. Greuel, P.T. Kreain and R.M. Bass, "Averaging models of PWM converter operating in discontinuous mode", *IEEE Transactions on Power Electronics*, vol. 16, pp. 482-491, July 2001.
- [73] A. Reatti and M. K. Kazimierzuk, "Small-signal model of PWM converters for discontinuous conduction mode and its application for boost converter", *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 50, pp. 65-73, January 2003.

CHAPTER 15**Symbolic Characterization of VCOs and its Application to Optimization Based Design****M. Helena Fino* and Fernando V. Coito***Instituto de Telecomunicações, Lisboa Portugal, FCT/UNL, Portugal*

Abstract: This chapter addresses the application of symbolic techniques on the characterization of ring Voltage Controlled Oscillators (VCO). The symbolic characterization of the VCOs comprises both the frequency-control voltage response and a simplified formula for evaluating the phase-noise in the oscillator. Two methodologies are shown for generating the frequency-control voltage response. The first one is based on the evaluation of the delay introduced by each VCO delay cell. In the second an approximate expression for the equivalent resistance of each VCO delay cell load is considered, and used for deriving the VCO model. The adoption of the proposed methodologies to submicron transistor sizes is illustrated. The application of the VCO characterization into an optimization based design is described.

Keywords: Ring VCO, VCO model, symbolic characterization, optimization-based design, deep-submicron VCO characterization, semi-symbolic methodology, phase-noise characterization, Maneatis delay cell, equivalent resistance, phase-noise minimization.

1. Introduction

The rapidly growing demand for lower cost and higher bandwidth as well as functionality of RF transceivers has motivated higher levels of integration of wireless communications systems operating at higher frequencies [1, 2]. Among the wireless transceiver building blocks Voltage Controlled Oscillators (VCOs) are a major concern for designers since they are responsible for the purity of the signal generated. Ring-VCOs are known for being easily integrated in standard CMOS technologies. They also show a wide tuning range and if implemented with a small number of stages they occupy less area than LC-VCOS. LC-VCOs on the other hand, show much better phase-noise behavior. This has motivated the development of research work regarding the design of low phase-noise ring-VCOs.

In spite of the widespread use both in communication circuits and in microprocessors, ring VCOs are usually designed empirically. Traditionally the design of a VCO starts with the choice of the number of stages for a desired oscillation frequency. Then a refinement of the results is performed through iterative simulations. This approach, however, is a time consuming prohibitive process because transient circuit simulations must be run long enough before steady state is attained. With the aim of increasing efficiency in the design process, accurate models for the evaluation of the delay introduced by each stage have been proposed in the literature [3, 4].

The necessity for designing low phase-noise ring-oscillators has motivated the adoption of design methodologies where not only frequency-control voltage is a major concern, but the minimization of the phase noise must also be accounted for. Nowadays, numerical simulators also offer the possibility for simulating time jitter or phase noise in the oscillation. Yet, the use of a numerical simulator, although producing accurate results, does not give the designer a qualitative insight into which

*Address correspondence to **Maria H. Fino**: Instituto de Telecomunicações, Lisboa Portugal, FCT/UNL, Portugal; E-mail: hfino@fct.unl.pt

parameters should be altered in order to improve the circuit behavior. A symbolic approach for the characterization of both frequency-control voltage and phase-noise must be adopted as a way of offering designers qualitative insight into the key design parameters, so that ring oscillators may be designed first time right without guess work and lengthy simulations [4].

In this chapter a brief description of CMOS ring VCOs will be presented, where particular emphasis for differential delay cell ring VCOs will be given. Then the symbolic characterization of the VCOs will be presented. The approach proposed relies on the resolution of the differential equations modeling each VCO delay cell. For the characterization of the phase noise/jitter the methodology proposed in [5] will be considered. A working example considering a symmetric load ring VCO [6] will be presented, and results obtained will be compared against those from numerical simulation. The symbolic characterization of the VCO will be used for integrating a phase-noise-aware optimization-based design procedure. A final example guiding the integration of the symbolic characterization into an optimization based design procedure will be presented. In the last section conclusions are offered.

2. Voltage Controlled Ring Oscillators

For the implementation of ring VCOs structures consisting of N inverters or delay cells with the output fed to the input are usually used. These oscillators are known for generating precise delays due to their inherently high delay linearity, *i.e.*, considering that all buffer stages are identical, the relationship between a buffer delay and the period of the signal generated is set by the number of stages [1].

2.1. Ring VCO frequency-voltage response

For the sake of modeling we start by considering each delay cell consisting of a negative conductance, $-G_m$, driving an RC load as illustrated in **Fig. 1**.

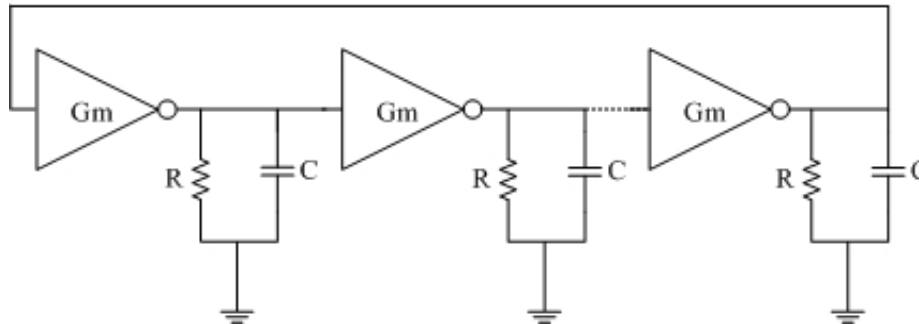


Figure 1: Ring oscillator structure.

For this case, the transfer function of each delay stage is given by

$$G(j\omega) = \frac{-G_m R}{1 + j\omega RC} \quad (1)$$

yielding to the N -stage ring oscillator closed-loop transfer function in

$$H(j\omega) = \left(\frac{-G_m R}{1 + j\omega RC} \right)^N \quad (2)$$

that can be rewritten as:

$$H(j\omega) = \left(\frac{GmR}{\sqrt{1+(\omega RC)^2}} \right)^N \exp(jN \operatorname{atan}(\omega RC)) \quad (3)$$

According to linear system theory and Barkhausen criteria, the frequency of oscillation, ω_{osc} , is given by:

$$\omega_{osc} = \frac{1}{RC} \tan\left(\frac{\pi}{N}\right) \quad (4)$$

yielding

$$f_{osc} = \frac{1}{2\pi RC} \tan\left(\frac{\pi}{N}\right) \quad (5)$$

that can be written as

$$f_{osc} = \frac{1}{2NRC} \left(\frac{\tan\left(\frac{\pi}{N}\right)}{\left(\frac{\pi}{N}\right)} \right) \quad (6)$$

where a minimum gain per stage, GmR is needed

$$GmR = \sec\left(\frac{\pi}{N}\right) \quad (7)$$

Another approach for calculating the frequency of oscillation of ring oscillators considers

$$f_{osc} = \frac{1}{2Nt_{delay}} \quad (8)$$

so we may conclude, from (6) and (7) that the delay introduced by each delay cell may be given by

$$t_{delay} = RC \frac{\left(\frac{\pi}{N}\right)}{\tan\left(\frac{\pi}{N}\right)} \quad (9)$$

We should note, that the second term in (9), is approximately unity for $N \geq 7$, yielding to

$$t_{delay} = RC \quad (10)$$

which is usually used saturated ring oscillators.

2.2. Phase noise in ring VCOs

Ideally an oscillator will exhibit a frequency spectrum consisting of a single impulse at the frequency of oscillation. Yet, due to noise causing variations in the phase of the output signal, the waveform of a real oscillator can be written as

$$v_o = A \cos[\omega_o t + \phi_n(t)] \quad (11)$$

where $\phi_n(t)$ is the phase noise of the oscillator.

The characterization of phase noise in ring oscillators has been carefully presented in [7]. However, for the sake of simplicity, we will follow the approach presented in [4], where the oscillator is considered as a linear system. This assumption leads to quite accurate results for ring oscillators with a small number of delay stages (typically up to five stages), since the output signal is approximately sinusoidal. For those oscillators with a larger number of delay stages, results obtained are less accurate.

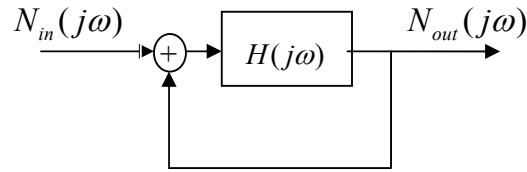


Figure 2: Feedback model for the ring oscillator.

Considering the feedback model of the ring oscillator as illustrated in **Fig. 2**, where $N_{in}(j\omega)$ and $N_{out}(j\omega)$ are the input and output noise signals, respectively, we obtain

$$\frac{N_{out}(j\omega)}{N_{in}(j\omega)} = \frac{1}{1-H(j\omega)} \quad (12)$$

Using the truncated Taylor series expansion of $H(j\omega)$ given by

$$H(j\omega) = H(j\omega_0) + \Delta\omega \frac{dH}{d\omega} \quad (13)$$

where ω_0 is the frequency of oscillation and imposing $H(j\omega_0) = 1$ in (12) the noise power will be given by

$$\left| \frac{N_{out}(j\omega)}{N_{in}(j\omega)} \right|^2 = \frac{1}{(\Delta\omega)^2 \left| \frac{dH}{d\omega} \right|^2} \quad (14)$$

Considering the expression for $H(j\omega)$ in (2) we obtain

$$\frac{dH}{d\omega} = -N(-GmR)^N \left(\frac{1}{1 + j \tan\left(\frac{\pi}{N}\right) \frac{\omega}{\omega_{osc}}} \right)^{(N+1)} \left(j \tan\left(\frac{\pi}{N}\right) \frac{1}{\omega_{osc}} \right) \quad (15)$$

Using Gm given by (7), we obtain

$$\left| \frac{dH}{d\omega} \right|_{\omega=\omega_{osc}}^2 = \left(N \sin\left(\frac{\pi}{N}\right) \frac{1}{\omega_{osc}} \right)^2 \quad (16)$$

yielding

$$\left| \frac{N_{out}(j\omega)}{N_{in}(j\omega)} \right|_{\omega=\omega_{osc}}^2 = \frac{\omega_{osc}^2}{(\Delta\omega)^2 N^2 \sin^2\left(\frac{\pi}{N}\right)} \quad (17)$$

For the evaluation of the input noise signal, input current noise sources are added to each oscillator delay stage, as illustrated in **Fig. 3**.

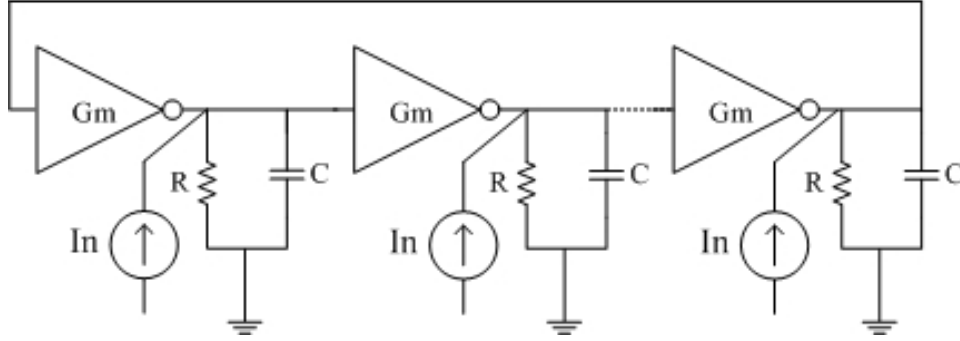


Figure 3: Ring oscillator with input current noise.

In this case the input noise signal is given by

$$N_{in}(j\omega) = i_n \frac{R}{(1 + j\omega RC)} \quad (18)$$

and

$$\left| N_{in}(j\omega) \right|_{\omega=\omega_{osc}}^2 = i_n^2 \frac{R^2}{1 + \tan^2\left(\frac{\pi}{N}\right)} \quad (19)$$

Considering that each of the N delay stages contributes with one the input current noise, we obtain

$$\left| \frac{N_{out}(j\omega)}{i_n(j\omega)} \right|_{\omega=\omega_{osc}}^2 = \frac{R^2 \omega_{osc}^2}{\left(1 + \tan^2\left(\frac{\pi}{N}\right)\right) N (\Delta\omega)^2 \sin^2\left(\frac{\pi}{N}\right)} \quad (20)$$

Finally, if we consider as in [4]

$$i_n^2 = 8k \frac{T}{R} \quad (21)$$

The total output noise is

$$\left| N_{out}(j\omega) \right|_{\omega=\omega_{osc}}^2 = \frac{R 8kT}{\left(1 + \tan^2\left(\frac{\pi}{N}\right)\right) N \sin^2\left(\frac{\pi}{N}\right)} \left(\frac{\omega_{osc}}{\Delta\omega} \right)^2 \quad (22)$$

Therefore, the phase noise will be equal to

$$PN(\Delta\omega) = \frac{R 8kT}{v_{osc}^2 \left(1 + \tan^2\left(\frac{\pi}{N}\right)\right) N \sin^2\left(\frac{\pi}{N}\right)} \left(\frac{\omega_{osc}}{\Delta\omega} \right)^2 \quad (23)$$

or

$$PN(\Delta\omega) = \frac{\tan\left(\frac{\pi}{N}\right)8kT}{v_{osc}^2 \omega_{osc} C \left(1 + \tan^2\left(\frac{\pi}{N}\right)\right) N \sin^2\left(\frac{\pi}{N}\right)} \left(\frac{\omega_{osc}}{\Delta\omega}\right)^2 \quad (24)$$

For the particular case of a three stage oscillator we obtain

$$PN(\Delta\omega) = \frac{8kT\sqrt{3}}{9v_{osc}^2 C \omega_{osc}} \left(\frac{\omega_{osc}}{\Delta\omega}\right)^2 \quad (25)$$

3. Symbolic VCO Frequency-Voltage Response

In this section ring VCOs comprising differential gain stages with symmetric loads as illustrated in **Fig. 4** will be considered [6].

The analytical expression for evaluating the delay introduced by each stage may be obtained considering the large signal behavior of the differential inverting stage. In this delay cell the current I_{bias} is regulated such that the output voltage swing is equal to the control voltage, V_C . For the sake of simplicity, we will assume that the current through the delay cell switches instantly from the left to the right branch after the differential input voltage, $(V_{in+} - V_{in-})$, changes polarity. The voltage V_1 , across capacitor C_1 , starts at 0V and approaches V_C , while the voltage V_2 , across capacitor C_2 , starts at V_C and approaches 0V. The next delay cell will start switching when $V_1 = V_2 = V_C/2$. The delay time is then given by the time spent for charging the capacitance to $V_C/2$ [3].

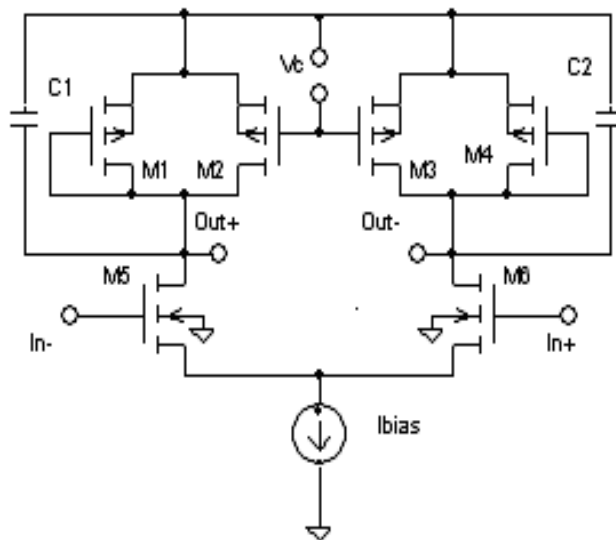


Figure 4: One of N -delay stage VCO.

This time may be evaluated considering the symmetric load cell represented in **Fig.5**.

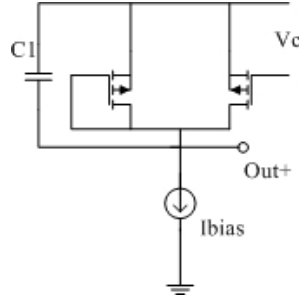


Figure 5: Symmetric load for the VCO differential delay stage.

For evaluating the delay time for charging for the capacitor to $V_c/2$, we must solve

$$C \frac{dV}{dt} + I_{sl}(V, V_c) = I_{bias} \quad (26)$$

where I_{sl} represents the sum of the load transistors currents. According to the different MOSFET transistor operating regions two major cases must be considered, as illustrated in **Table 1**

Table 1: Symmetric Load MOSFET Operating Regions.

	Operating Regions	M2	M1
$V_c < 2V_t$	$0 < V_{ds} < V_c - V_t$	Triode	Off
	$V_c - V_t < V_{ds} < V_t$	Saturation	Off
	$V_t < V_{ds} < V_c$	Saturation	Saturation
$V_c \geq 2V_t$	$0 < V_{ds} < V_t$	Triode	Off
	$V_t < V_{ds} < V_c - V_t$	Triode	Saturation
	$V_c - V_t < V_{ds} < V_c$	Saturation	Saturation

In this chapter we will address the case for $V_c > 2V_t$, where two regions must be considered. In region I while $V < V_t$, transistor M_1 is off and transistor M_2 is in triode, and (25) becomes

$$C \cdot \frac{dV}{dt} + \beta \cdot \left((V_c - V_t)V - \frac{V^2}{2} \right) = \beta \cdot (V_c - V_t)^2 \quad (27)$$

With initial conditions given by

$$V(0) = 0 \quad (28)$$

A second region must be considered for $V > V_t$, where transistor M_1 starts conducting while transistor M_2 is in triode. Here, (26) becomes

$$C \cdot \frac{dV}{dt} + \beta \cdot \left((V_c - 2V_t)V + \frac{V_t^2}{2} \right) = \beta \cdot (V_c - V_t)^2 \quad (29)$$

With initial conditions given by

$$V(0) = V_t \quad (30)$$

The overall delay time will be given by the addition of the time, t_1 , for V to reach V_t in region I, with the time, t_2 , for V to reach $V_c/2$, in region II.

In the next sub-sections, results obtained with the *Matlab* symbolic toolbox will be presented. In the first case, symbolic expressions for evaluating the delay introduced by each symmetric-load delay element are obtained. Then, an approximate expression for characterizing the symmetric load equivalent resistor is presented. This result will be used for obtaining an approximate symbolic characterization of the VCO frequency *versus* control voltage response. Finally, limitations of the proposed methodologies on the characterization of VCOs using sub-micron technologies are pointed out and solutions proposed.

3.1. Symbolic characterization of the symmetric load delay cell

The generation of the symbolic solution of the equations leading to the evaluation of the delay time of each VCO stage has been implemented with the *Matlab* symbolic toolbox. In **Fig.6** a caption of the code for evaluating the overall delay time for the symmetric load capacitor attaining voltage $V_c/2$ is represented.

```
function varargout=calcdelay(X)

syms v Beta Vc Vt ibias ils itot
% region 1- M1 off and M2 triode
ils=Beta*((Vc-Vt)*v-0.5*v^2);
ibias=Beta*(Vc-Vt)^2;
dv1m=strcat('Dv+',char(ils),'=',char(ibias));
v1m=simple((dsolve(dv1m,'v(0)=0.0')));
x1m=v1m-Vt;
t1m=simple(C*solve(x1m,'t'));
pretty(t1m)
%region 2 -M1 saturated M2 in Triode
ils=simple(Beta*((Vc-Vt)*v-
0.5*v^2)+0.5*Beta*(v-Vt)^2)
ibias=B*(Vc-Vt)^2;
dv2m=strcat('Dv+',char(ils),'=',char(ibias));
v2m=simple((dsolve(dv2m,'v(0)=Vt')));
x2m=v2m-Vc/2;
t2m=simple(C*solve(x2m,'t'));
pretty(t2m)
tdelay=simple((t1m+t2m));
```

Figure 6: Matlab code for delay time evaluation.

The results obtained with *Matlab* are,

$$t_1 = \frac{C}{B} \frac{\frac{\pi}{2} - 2 \operatorname{atan} \left(\frac{V_c - V_t}{V_c - 2V_t} \right)}{(V_c - V_t)} \quad (31)$$

and

$$t_2 = \frac{C}{B} \left(\frac{1}{(V_c - 2V_t)} \log \left(1 + \frac{(V_c - 2V_t)^2}{(V_c - V_t)^2} \right) \right) \quad (32)$$

Leading to the overall delay time given by

$$t_{delay} = \frac{C}{B} \left(\frac{\frac{\pi}{2} - 2 \operatorname{atan} \left(\frac{V_c - V_t}{V_c - 2V_t} \right)}{(V_c - V_t)} + \frac{C}{B} \left(\frac{1}{(V_c - 2V_t)} \log \left(1 + \frac{(V_c - 2V_t)^2}{(V_c - V_t)^2} \right) \right) \right) \quad (33)$$

This equation may be integrated into (8) for obtaining the frequency-to-control voltage symbolic characterization of the VCO.

Yet, this approach, although producing accurate results, is much too complex for offering the designer qualitative insight into which are the key parameters responsible for the VCO frequency response. With the aim of obtaining a more intuitive characterization of the VCO frequency response a second approach may be used where an approximate symbolic expression is obtained.

3.2. Deriving an approximate symbolic characterization of the VCO

Having in mind the general VCO structure represented in **Fig.1**, a simplified frequency to voltage expression may be obtained, based on the generation of an approximate expression for the equivalent resistance of the delay cell symmetric load.

An approximate expression for the equivalent resistance of the symmetric load may be obtained if we consider the voltage-to-current relation of the symmetric load represented in **Fig. 7**.

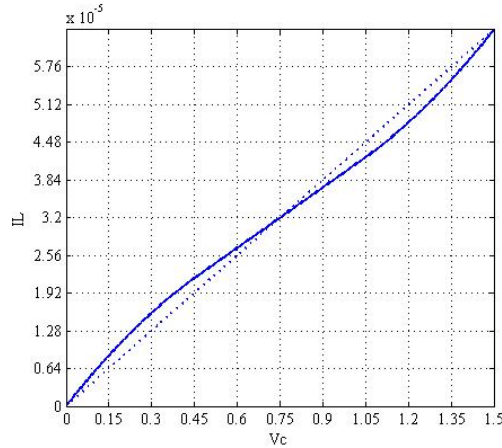


Figure 7: Current *versus* voltage of a symmetric- load.

In the same figure, a dashed line represents an approximate behavior of the voltage-to-current relation which leads to an equivalent resistance given by the ratio between the control voltage and the maximum current, i.e.,

$$R_{eq} = \frac{V_c}{\beta(V_c - V_t)^2} \quad (34)$$

Using Matlab symbolic toolbox for solving (26) with

$$I_{sl} = \frac{V}{\text{Re } q} \quad (35)$$

the approximate expression for evaluating the delay introduced by each symmetric load delay element becomes

$$t_{\text{delay}_{-ap}} = \frac{C}{\beta} \left(\frac{1}{(V_c - 2V_t)} \log \left(1 + \frac{V_c^2 - 2V_c V_t}{(V_c - V_t)^2} \right) \right) \quad (36)$$

A more intuitive result may be obtained by considering (34) and (6) yielding the approximate frequency-control voltage equation

$$f_{\text{osc}} = \frac{\beta(V_c - V_t)^2 \tan\left(\frac{\pi}{N}\right)}{2NCV_c \left(\frac{\pi}{N}\right)} \quad (37)$$

That for the particular cases where $N \geq 7$, (37) can be simplified to

$$f_{\text{osc}} = \frac{\beta(V_c - V_t)^2}{2NCV_c} \quad (38)$$

This is the same expression which is proposed in [3] for evaluating the frequency of oscillation for a given control voltage.

3.3. VCO characterization in deep-submicron technologies

As we have mentioned, the symbolic characterization proposed in the previous section relies thoroughly on the *DC* characterization of the MOSFET transistors comprising the symmetric loads of the VCO delay stages. Unfortunately, as the sizes of MOSFET transistors scale down, the quadratic model is no longer applicable, for new physical effects such as the velocity saturation and mobility degradation, among others, must be accounted for. To overcome the inaccuracy of Shockley's MOSFET model when applied to submicron circuits, Sakurai's *Npower* model was proposed [8]. This model has been successfully used in the derivation of analytical expressions for evaluating delay and power dissipation in submicron CMOS gates [9, 10]. The main points to be taken into account for our work are:

$$I_{\text{Dsat}} = B(W/L_{\text{eff}})(V_{\text{GS}} - V_t)^n \quad (39)$$

$$I_D = I_{\text{Dsat}}(1 + \lambda V_{\text{DS}}) \Leftarrow V_{\text{DS}} \geq V_{\text{Dsat}} \quad (40)$$

where L_{eff} is the effective transistor length. As we may easily conclude from (39) the dependency of I_D on V_{GS} is no longer quadratic. The power is n (usually ranging from 1.5 to 1.8, depending on the technology) as stated in (39). This has severe implications on the methodology proposed in the previous section. In this case, the differential equation modeling the charging process of the capacitances will have to deal with equations for modeling the currents according to *Npower* model. Since the MOSFET currents depend on the control voltage with a non-integer power, the symbolic solution of the equations is no longer easily obtained. In this case only a semi-symbolic solution can be obtained.

As an example a caption of the code for evaluating the delay time, t_d , for the symmetric load capacitor to charge to voltage V_t , is represented in **Fig.8** where transistors with $W/L_{ef} = 5/.4665$, and the model parameters in **Table 2**, were considered.

```
function x=numsoldelay ()
global Vdd B Vc n Vt C K m lambda
W=5;
L=.4665;
B=21.276e-6*W/L;
Vt=0.462;
Vc=1.2;
C=0.75*5e-15*(5*0.5+50*0.66)
K=.828;
m=.770;
Vdd=1.8;
lambda=0.053
n=2%1.6;
[t,y]=ode45(@getI1,[0 .4e],[0.0], ..
odeset('reltol',1e-12));
td=interp1(y,t,Vth)
end

function z=getI1(t,V)
global Vdd B Vc n Vth C K m lambda
Ils=B*((Vc-Vt)^n)*(1+lambda*V);
vdsat=K*(Vc-Vt)^m;
rat=V/vdsat;
Ibias=2*B*((Vc-Vt)^n)*(1+lambda*V);
if rat<1
    Ils=Ils*(2-rat)*rat;
end
z=(-Ils+Ibias)/C;
end
```

Figure 8: Matlab code for delay time evaluation with *Npower* model.

Table 2: *Npower* Model Parameters.

Parameters	Typical Case
Vdd	1.8
B	21.276e-6
n	1.60
K	.828
m	.770
λ	.053
V_{th}	.462

A delay time of $t_1 = 3.25e-10$ s was obtained for $n = 1.6$. In the case of $n = 2.0$ (quadratic model of the MOS transistor), a delay time of delay time of $t_1 = 3.45e-010$ s is computed, against a delay of $t_1 = 3.38e-010$ s obtained with the symbolic approach in Section 3. The minimal difference between the two values may be due to the fact that

in the symbolic solution we did not consider the variation of the drain current, I_D , with the drain to source to voltage, V_{DS} .

On what concerns the approximate characterization of the VCO, the expression for the equivalent symmetric-load resistance becomes

$$R_{eq} = \frac{V_c}{2B(W/L_{eff})(V_c - V_t)^n (1 + \lambda_o V_c)} \quad (41)$$

Leading to the frequency *versus* control voltage given by

$$f_{osc} = \left(\frac{B(W/L_{eff})(V_c - V_t)^n (1 + \lambda_o V_c)}{V_c N C_{eff}} \right) \left(\frac{\tan(\pi/N)}{(\pi/N)} \right) \quad (42)$$

In **Fig. 9** a comparison between the frequency response of a 7-stage VCO, obtained with (42) and results from simulation with electrical simulator *Hspice* are represented.

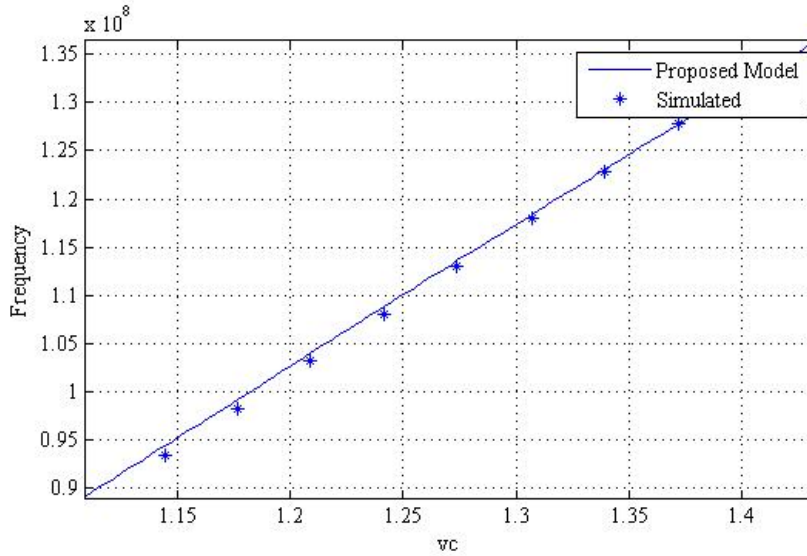


Figure 9: Frequency to control voltage response of a 7-delay stage VCO.

From the results represented we may conclude on the high accuracy of the results obtained with the symbolic approximate expression derived.

4. Optimization Based VCO Design

The main requirements for integrated VCO design include the center frequency, the tuning range, the power supply voltage, power consumption and phase-noise specifications [11].

The design of VCOs usually starts with the choice of the number of stages for a desired oscillation frequency. Then a refinement of the results is performed through iterative simulations. Since transient simulations must be run this approach has severe limitation concerning the efficiency of the methodology. To overcome this problem, accurate models for the evaluation of the delay introduced by each stage have been proposed in the literature [3, 4]. These models may be integrated in optimization environments, but the process is still not efficient since the search for solutions does not take into account qualitative knowledge regarding the VCO key elements responsible for required characteristics. So a huge number of intermediate design solutions are generated which are discarded.

By using the symbolic characterization of the VCO, a methodology may be defined where a prioritization of the design may be used and constraints on components sizes may be imposed, preventing the generation of unsatisfactory intermediate solutions and thus increasing the efficiency of the design process.

4.1. Working example

A working example will be used for illustrating the use of the qualitative knowledge acquired from the symbolic characterization of ring VCOs in a more efficient optimization based design methodology. In this example the design of a ring VCO working with a central frequency of 1.0 GHz and showing a phase-noise of -100 dBc/Hz at a 1 MHz offset is envisaged. For the implementation of this ring VCO a technology with a supply voltage of 1.8V will be chosen.

If we consider a control voltage of 1.0V for a frequency of 1 GHz, than the output voltage will be $0.35V_{rms}$. Since the phase-noise characterization has been derived considering a linear oscillator, we will choose a three-stage VCO topology. In this case, using (25) we obtain

$$PN(\Delta\omega) = \frac{8}{9} \frac{410^{-21} \sqrt{3} 2\pi 10^9}{0.35^2 C (2\pi 10^6)^2} \quad (43)$$

Yielding, for a phase noise of -100 dBc , *i.e.*, 10^{-10}

$$C \geq 80\text{ fF} \quad (44)$$

For transistor sizing, an optimization methodology may be employed where (42) is used for obtaining the symmetric load transistor sizes. Here a minimum value for transistor length of 1.5 times the minimum allowed for the technology chosen should be imposed. Concerning the differential pair transistor sizes (switches) constraints must be imposed so that the condition for oscillation is guaranteed, *i.e.*, using (7) and (41),

$$Gm \geq \left(\frac{4 B_{load} (W_{load} / L_{eff_load}) (V_c - V_{Th})^n (1 + \lambda_o V_c)}{V_c} \right) \quad (45)$$

Once transistor sizes are obtained, the equivalent Capacitance may be obtained with

$$C_{eff} = 0.5 C_{ox} (W_{switch} L_{switch} + W_{load} L_{load}) \quad (46)$$

Should this Capacitance value be smaller than the minimum allowed, an external capacitor is added to each delay cell.

5. Conclusion

In this paper a symbolic approach for the characterization of Ring VCOs was introduced. This characterization entails information regarding not only the evaluation of the frequency of oscillation of a ring VCO, but an approximate analytical expression for evaluating the VCO phase-noise. In the work presented the analytical characterization of the frequency of oscillation is obtained based on the delay introduced by each VCO delay element. Two approaches have been shown, for the automatic generation of the symbolic characterization of the delay introduced by

each VCO element. In the first approach an exact formula is obtained while in the last approach an approximate, yet simple and accurate expression is obtained. However, both results obtained show severe inaccuracy when submicron technologies are used. To overcome this problem a semi-symbolic methodology is proposed, showing quite accurate results. Finally, the use of the symbolic characterization for the development of an optimization based methodology for designing ring VCOs is illustrated.

References

- [1] B. Razavi, *Design of analog CMOS integrated circuits*, McGraw-Hill Series in Electrical and Computer Engineering”, Boston 2000.
- [2] T. Tsang, M. El-Gamal, Krzysztof, *et al.*, “Current status of CMOS low voltage and low power wireless IC designs”, *Analog Integrated Circuits and Signal Processing*, Springer, pp. 9-18, January, 2007.
- [3] T. Toifl, “Integrated circuits for the synchronization of high-energy physics Experiments”, Ph.D. Dissertation, Technische Universitat Wien, Austria, 1999.
- [4] M.H. Fino, A.B. Leal, “Accurate modeling of submicron symmetric load ring VCOs”, in *IEEE International Conference on Electronic Design*, 2004, pp. 1024-1027.
- [5] J.W. Rogers, C. Plett, and Foster Dai, *Integrated circuit design for high-speed frequency synthesis*, Artech House in 2006.
- [6] J.G. Maneatis, “Low-jitter process-independent DLL and PLL based on self-biased techniques”, *IEEE Journal of Solid State Circuits*, vol. 31, no. 11, pp.1723-1732, November 1996.
- [7] J.A. McNeill, D. Ricketts, *The designer's guide to jitter in ring oscillators*, Springer 2009.
- [8] T. Sakurai, A. R. Newton, “A simple MOSFET model for circuit analysis”, *IEEE Transactions on Electron Devices*, vol. 38, no. 4, pp. 887-893, April 1991.
- [9] L. Bisdounis, S. Nikolaidis, O. Koufopavlou, “Analytical model for CMOS short-circuit power dissipation”, *Integrated Computer-Aided Engineering Journal, special issue on low-power electronic systems*, vol. 5, no. 2, , pp. 129-140, April 1998.
- [10] J.L. Rossello, J.Segura, “Charge-based analytical model for the evaluation of power consumption in sub-micrometer CMOS buffers”, *IEEE Transactions on Computer Aided Design*, vol.21, no. 4, pp. 433-448, April 2002.
- [11] M. Tiebout, *Low power VCO design in CMOS*, Springer Series in Advanced Microelectronics, 2006.

AMS Synthesis Using Symbolic Methods

Mauro Santos and Nuno Horta *

Instituto Superior Técnico, Instituto de Telecomunicações, Lisbon, Portugal

Abstract: This chapter addresses the problem of automatically generating data converter topologies, from algorithm descriptions to behavior building blocks, using a symbolic synthesis methodology. The discussed approach consists of an algorithm-driven methodology, which employs a combination of symbolic signal flow graph techniques, to generate canonical representations for data converter algorithm descriptions, together with pattern recognition techniques, to determine the appropriate functional building blocks for the data converter topology. The methodology is illustrated by working examples where VERILOG-AMS descriptions are considered at the input stage, for algorithm specification, and at the output stage, for topology description, in both cases the CADENCE® IC Design Environment is used for validation purposes.

Keywords: Symbolic synthesis, algorithm-driven methodology, data converters, topology generation, algorithm descriptions, signal flow graphs, canonical representations, pattern recognition, behavior building blocks, functional building blocks, VERILOG-AMS.

1. Introduction

The advances in Very Large Scale Integration (VLSI) technologies combined with the market demand lead, more and more, to the integration of complete signal processing systems in a single chip and, also, to the spreading of Application Specific Integrated Circuits (ASICs). This conducts, naturally, to an increasing interest on the development of Computer Aided Design (CAD) and Design Automation (DA) tools to support integrated circuit (IC) design and, specially, the design of integrated data conversion systems due to their broad range of applications [1, 2].

In the later 80's, the first tools to approach the synthesis of data converters made it by implementing an architecture-constrained methodology, together with the use of standard cells to generate a semi-custom layout [3-5]. This approach applies only to very specific cases due to its high dependence on lower level circuitry. In the early 90's, the evolution of lower level tools together with the more systematic application of hierarchical concepts led to an increased flexibility of design automation by specifying the converter at the topological level and using lower level tools to generate the appropriate sub-blocks [6-13]. Meanwhile, several analog design automation tools were developed using the most promising computation techniques to explore and optimize analog IC design [14-42]. Despite, the evolution higher level CAD systems are still limited by the lack of formal synthesis techniques available for analog and mixed-signal design.

This chapter discusses a synthesis methodology, based on symbolic methods, which implements a general synthesis approach covering a wide range of data converters requirements (linear and non-linear). Particularly, a methodology for the automatic synthesis of data conversion systems from the algorithm description to the functional

*Address correspondence to Nuno Horta: Instituto Superior Técnico, Instituto de Telecomunicações, Lisbon, Portugal; E-mail: nuno.horta@lx.it.pt

topology definition and sub-blocks specification is presented. The implemented symbolic approach employs Signal Flow Graph (SFG) descriptions together with pattern recognition techniques to allow the synthesis process to move to higher levels of abstraction, thus providing a computer-based framework for the systematic and uniform treatment of new conversion algorithms and/or new implementation topologies.

2. Symbolic Synthesis Proposed Approach: Overview

The described approach, illustrated in **Fig. 1**, was implemented on a C/PROLOG environment incorporating a sequence of symbolic and numerical transformations in order to map the data converter algorithms into feasible electrical topologies. The implemented methodology consists of three different phases, along three hierarchical levels, with specific interfaces for simulation within CADENCE[®] IC Design Environment [43]. The first phase is decomposed into two parts. First, the acquisition of the system requirements, *i.e.*, the desired algorithm description, made with VERILOG-AMS [44], and the performance specs. Then, the validation of the algorithm description is carried out by simulation. Next, an equivalent description is achieved through the interpretation, in PROLOG, of the algorithm description primitives, in VERILOG-AMS, which are translated into SFG substructures, generating a file with extension *gra*. After that, another symbolic task is performed, in PROLOG, for the simplification and partitioning of the previously achieved graph, originating a file with extension *sag* including the graph in a canonical form. The second phase consists of the functional topology generation. In this phase, a library of functional blocks is described in terms of SFG structures, in order, to allow the topology generation process to act as a pattern recognition task over the graph, in PROLOG, and originating the file with extension *top* containing the topology description. The topology validation is performed through simulation using functional block models described in VERILOG-AMS.

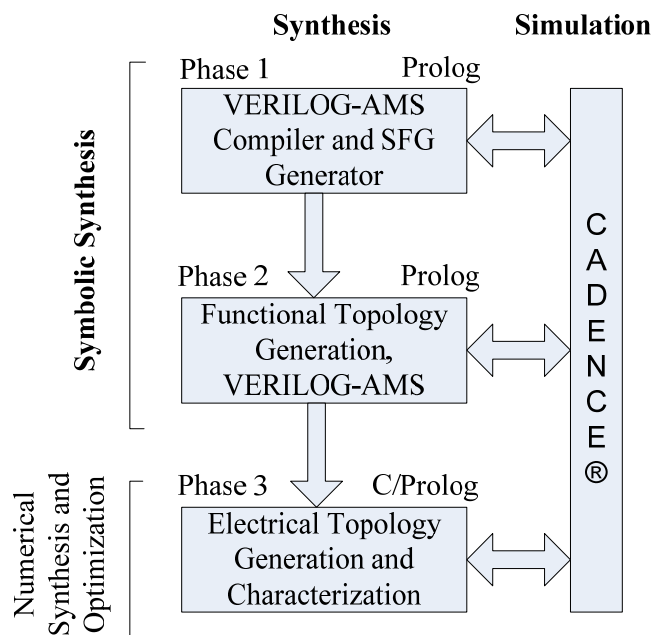


Figure 1: AMS synthesis approach.

The third phase consists of the electrical topology generation, *i.e.*, electrical sub-blocks selection, and sub-block specs generation. In this phase, although the topology generation follows a similar process to the previous phase, the generation of the specs for the sub-blocks was implemented using both PROLOG and C. Here, another library was developed to aid the task of identifying electrical structures. In this case, the generated topologies are described in a file with extension *ele*, which is used as an entry for the sub-blocks specs generation task. These specs are obtained through the application of design rules over the generated topology and produce a file with extension *siz*, which includes the requirements for each of the sub-blocks. Finally, validation is carried out by electrical simulation. The achieved sub-block requirements are, then, submitted to lower level synthesis tools. In this chapter only symbolic processing phases, respectively, phase 1 and phase 2, are discussed.

3. HDL Model for Algorithm Descriptions

In order to describe data conversion algorithms for automatic synthesis it is mandatory to use a description language which supports both functional descriptions and the appropriate interaction between analog and digital signals. For this purpose VERILOG-AMS together with CADENCE[®] IC Design Environment were chosen to allow both a faster prototyping and a validation mechanism. The next paragraphs present the model structure and the subset of language primitives considered for implementing data converter algorithm descriptions.

3.1. Model structure

The algorithm description consists of a sequence of logically independent conversion time steps comprising the relevant mixed analog-digital signal operations associated with the conversion process. This is encapsulated into a functional model whose structure, illustrated in **Fig. 2**, includes three main parts, namely the model identification, the model declaration and the model function.

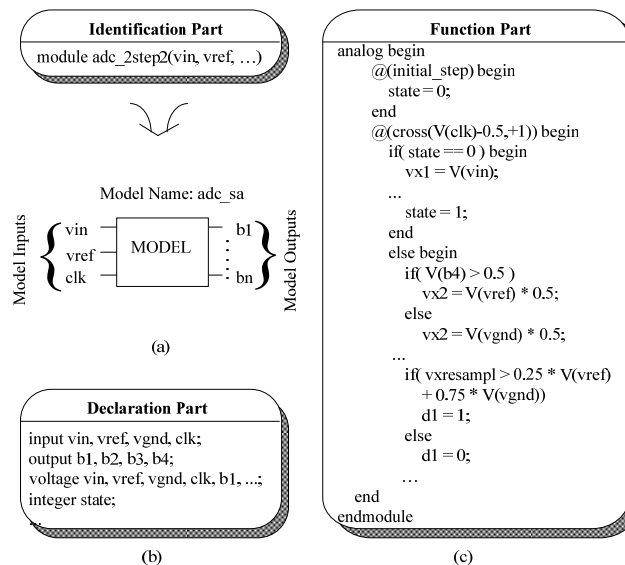


Figure 2: The main parts of the algorithm description structure are (a) model identification, (b) model declaration, (c) model function.

The *model identification* part in **Fig. 2(a)** represents the header of the functional model and includes the model name and the external interface pins. The *model declaration* part in **Fig. 2(b)** declares the type of each variable used to describe the

model inputs/outputs (external pins) and the different conversion steps. The variables are classified either as *input* and *output*, if they correspond to external pins, or as *integer* when they are used exclusively for describing the algorithm. A clear distinction between analog, state and digital signals is ensured by using *real*, *integer* and *reg* variable data types for classifying both the interface and the internal variables. Finally, the *model function* part in **Fig. 2(c)** describes the algorithm function by using a subset of statements which are similar to the ones used in most structured programming languages. The algorithm description is carried-out in the time domain and driven by the events occurring at each conversion cycle.

3.2. Language primitives

A subset of the language primitives is used to describe efficiently the operations and conditions between the analog and/or digital signals usually needed in conversion algorithms. The construct *if-else* allows the implementation of two fundamental types of conditions, the equality and the inequality, presented in **Table 1**. The equality is related to digital expressions, due to the nature of digital signals that have a well defined set of values, whereas the inequalities are associated with analog signals. Therefore, on the one hand, the digital conditions are applied to the state identification, encoders description and any other condition with only integer or logic variables while, on the other hand, the analog conditions determine whether a signal passed a specific threshold value corresponding to a change on the digital code or any other condition involving real variables. In summary, these conditions are mainly used to describe the A/D and D/A signal interfaces. The primitives/operators “<+” and “=” allow the description of signal processing both at the analog and digital levels, by executing operations over real and logic variables, respectively, as described in **Table 1**. The formats of other useful primitives used to define the operation steps or time slots, such as, *while*, *@*, *for*, are similar to the ones presented above. Examples of algorithm descriptions will be presented in section 5 when discussing the HDL to SFG translation.

Table 1: Description Primitives and Assumptions.

Description\Signal	Analog	Digital
Declaration	Real	Logic
Attribution	<p>Linear</p> $as < + \sum_i^n k_i a_i$ <p>Non-Linear</p> $as < + \prod_i^n k_i a_i$	$ds = f(d_{1..n})$
Condition	$IF a_i (>, <) a_j$	$IF f(d_{1..n})$
Event Control	<p>@ (event_expression)</p> <p>FOR (condition)</p> <p>WHILE(condition)</p>	
<p>a_s, a_i, a_j - analog signals; d_s, d_i, d_n - digital signals</p> <p>k_i - real coefficient</p>		

4. SFG Representation

In this section the proposed SFG representation is discussed, first, by reviewing the classic SFG approach and, then, by introducing a modified SFG representation to accommodate both analog and digital signals in a canonical form.

4.1. Signal flow graphs - traditional approach

The algorithm description requires the coexistence of analog and digital signals. The desired representation is attained by merging the traditional definitions and concepts, from the graph theory, with definitions and concepts developed for the specific case of mixed analog-digital systems. In the next subsections, a brief introduction to the traditional graph theory [45] is given and the most relevant definitions and concepts are presented as a foundation for understanding the developed modified SFG approach, which will be discussed in section 4.2.

4.1.1. Definitions and concepts

A SFG consists of a set of interconnected branches. The branch is an oriented line segment, as illustrated in **Fig. 3**, and the basic element for SFGs representations. The branch orientation, expressed by its arrow, indicates the signal flow direction. The branch terminals, designated by nodes, indicate the signal input and output, and are the linking points with other branches. The branch coefficient, designated by gain, indicates the output to input ratio.



Figure 3: SFG basic element.

In order to clearly illustrate the concepts in discussion, a graph, G , presented in **Fig. 4**, is used as reference. In this case, the graph is composed by nine linked branches based on four nodes. Therefore, the flowing signal is affected both by the chosen links and the nine existing coefficients.

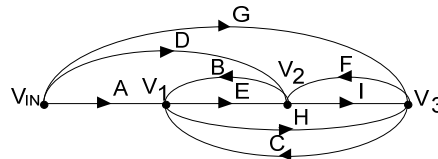


Figure 4: A signal flow graph.

Next, some of the more useful definitions for the SFG analysis are presented:

- **subgraph**: is a graph, G_i , composed by a subset of branches from the original graph, G .
- **input node**: is a node where at least one branch has the origin, but where no branch ends. These nodes are usually referred as sources, *e.g.* V_{IN} .
- **output node**: is a node where at least one branch ends, but where no branch has its origin. These nodes are usually referred as sinks.
- **incoming branch**: is the term given to the branch relatively to the node where it ends, *e.g.* branches A, B and C relatively to node V_1 .
- **outgoing branch**: is the term given to the branch relatively to the node where it has the origin, *e.g.* branches B and I relatively to node V_2 .
- **node degree**: is the total number of incoming and outgoing branches to a specific node, *e.g.* node V_1 has a degree of 5 in the original graph.
- **path**: is a subgraph with nodes of degree 2 with exception of the extremes having degree 1. The nodes sequences $V_{IN} V_1 V_2 V_3$ and $V_1 V_3 V_2$ are examples of paths, respectively, from V_{IN} to V_3 and from V_1 to V_2 .
- **forward path**: is a path where the nodes appear in a sequence from the input to the output or other node in analysis, *e.g.* $V_{IN} V_1 V_2$.
- **backward path**: is a path where the nodes appear in a sequence from the output

or other node in analysis to the input, e.g. $V_3 V_2 V_1$.

- **closed path**: is a path where the final node is identical to the initial node, e.g. $V_1 V_2 V_3 V_1$.
- **feedback path**: is the set of branches that compose a closed path.
- **forward node**: is a node that appears in a forward path.
- **feedback node**: is a node that appears in a feedback path.
- **forward branch**: is a branch that appears in a forward path.
- **feedback branch**: is a branch that appears in a feedback path.
- **forward graph**: is a graph composed only by forward branches.
- **feedback graph**: is a graph with at least one feedback node.

4.1.2. Signal flow graphs algebra

In the SFGs there are two implicit operations. The signal weighted sum, achieved by the convergence of several branches into a node and, the product of the signal by a coefficient, the gain, as illustrated on **Table 2**.

Table 2: SFG's Elementary Operations.

Operation	Graph	Expression
Sum		$x_i = \sum_{j=1}^n a_{ji} x_j$
Product		$x_i = \left(\prod_{j=1}^{i-1} a_j \right) x_{i-1}, i=1, 2, \dots, n$

The described operations allow the definition of elementary transformations, illustrated in **Table 3**, which are extremely useful for SFG manipulation. The first four transformations are derived directly from the algebraic operations properties, e.g., the serial association, the parallel association, the backward factorization and forward factorization, while the last two are achieved by algebraic manipulations, e.g., the self-loop and the star-mesh.

Table 3: SFG's Elementary Transformations.

Transfor.	Original Graph	Resulting Graph	Transfor.	Original Graph	Resulting Graph
Serial			Factorization I		
Parallel			Self-Loop		
Factorization I			Star-Mesh		

4.1.3. Description and analysis of equations systems by SFGs

In this subsection, the use of SFGs to solve symbolically a system of equations is described, in order to illustrate their potential, and to prepare the introduction of new concepts related to the simultaneous processing of analog and digital signals.

The graph description of an equations system is not unique, i.e., on the representation of each elementary term by branches, the use of different description strategies lead to different, but equivalent, graphs structures [45]. In the present case, the equations system representation will follow the Mason's SFG approach [46-47], as well as, the general formulation for the obtaining the solutions.

The SFG description of a three equations system, following the Mason's approach, is illustrated in Fig 5. The equations system described in the canonical form $AX = B$, is rewritten so that each variable is expressed in terms of the others. Then, each equation is translated into a SFG. Finally, the superposition of the achieved graphs corresponds to the complete graph representing the three equations system.

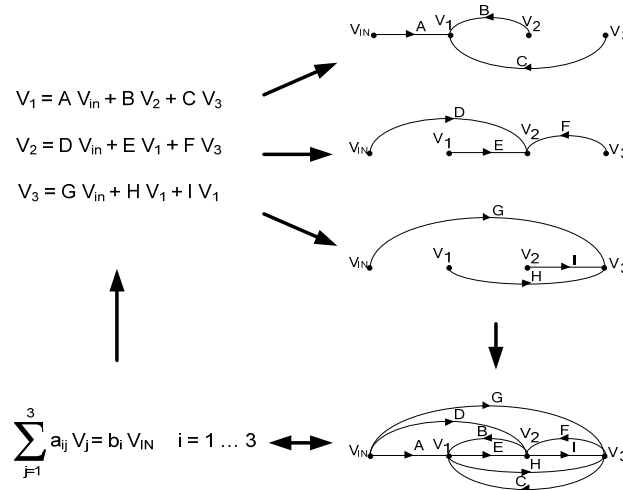


Figure 5: The SFG of a three equations system, following the Mason's approach.

The equivalence between the graph and the equations system means the possibility of achieving the system's solution by applying successively the elementary transformations until the graph is reduced to a single branch, between the input and each output, where the input node corresponds to the input signal V_{IN} , the output nodes correspond to variables being determined and the branch coefficient is the output to input ratio. The process of successive reduction of the graph is extremely laborious for graphs with a large number of branches, and so, limiting the applicability.

In order to solve this problem there are several solutions. Here, the Mason's rule is outlined as a general formulation to determine the gain between any two points in a SFG. The general expression for the gain, between any to graph nodes, is:

$$G = \frac{1}{\Delta} \sum_k G_k \Delta_k \tag{1}$$

where

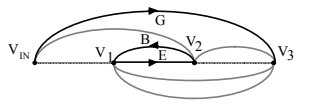
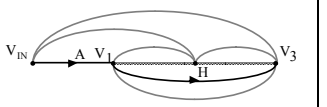
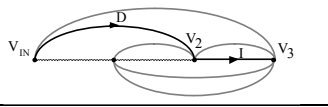
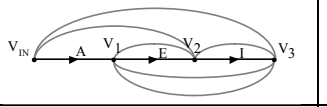
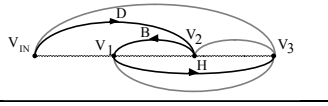
- G_k : represents the gain of the k^{th} forward path.
- Δ : designates the graph determinant and is given by

$$\Delta = 1 - \sum_m P_{m1} + \sum_m P_{m2} - \sum_m P_{m3} + \dots \tag{2}$$

- P_{mr} : corresponds to the product of m^{th} combination of r disjoint closed paths.
- Δ_k : is designated by cofactor of the forward path and corresponds to the value of Δ for the subgraph not touching on the forward path.

Next, as an example, the gain V_3/V_{IN} is determined. The evaluation starts by the identification of all the forward paths, between V_{IN} and V_3 , and the disjoint feedback paths between them and located in the subgraph not touching the forward path, as presented on Table 4.

Table 4: Forward Paths and Disjointed Feedback Paths.

Forward Paths and Feedback Paths	Gain G_k	Cofactor Δ_k	Forward Paths and Feedback Paths	Gain G_k	Cofactor Δ_k
	$G_1 = G$	$\Delta_1 = 1 - BE$		$G_4 = AH$	$\Delta_4 = 1$
	$G_2 = DI$	$\Delta_2 = 1$		$G_5 = AEI$	$\Delta_5 = 1$
	$G_3 = BDH$	$\Delta_3 = 1$			

After that, the evaluation continues by determining the gain of each forward path, G_k , and the gains of the feedback paths, having in mind the cofactors, Δ_k . In this case, the gain expression has the following numerator:

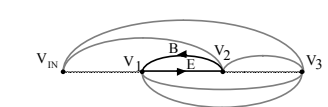
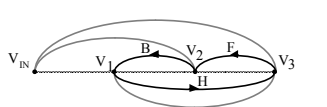
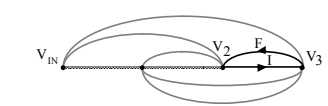
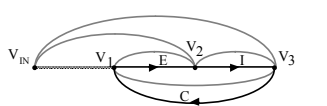
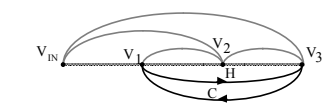
$$\sum_k G_k \Delta_k = G_1(1 - P_{11}) + G_2 + G_3 + G_4 + G_5 \tag{3}$$

this is,

$$\sum_k G_k \Delta_k = G(1 - BE) + DI + BDH + AH + AEI \tag{4}$$

The second phase on determining the gain expression refers to the evaluation of graph determinant, Δ , i.e, the expression denominator. Therefore, the disjointed feedback paths must be successively determined. In this case, only one path appears during each iteration, as presented on **Table 5**.

Table 5: Sets of Disjointed Feedback Paths.

Disjointed Feedback Paths	Gain	Disjointed Feedback Paths	Gain
	$P_{11} = BE$		$P_{41} = BFH$
	$P_{21} = FI$		$P_{51} = CEI$
	$P_{31} = CH$		

The appearance of various paths simultaneously would have implied the inclusion of another sum in the determinant expression. So, in this case the denominator is given by

$$\Delta = 1 - (P_{11} + P_{21} + P_{31} + P_{41} + P_{51}) \tag{5}$$

this is,

$$\Delta = 1 - (BE + FI + CH + BFH + CEI) \tag{6}$$

Finally, the expression of the gain is achieved and is given by

$$G = \frac{V_3}{V_{IN}} = \frac{G(1 - BE) + DI + BDH + AH + AEI}{1 - (BE + FI + CH + BFH + CEI)} \quad (7)$$

In the case of multiple inputs it is required the application of the Superposition Principle because the Mason's rule only relates the output to one entry. Another formulation for the gain evaluation problem is the Coates's Rule [48] which allows the direct determination of the relation even with multiple inputs. However, Coates's SFG approach is not based in a cause-effect relation, as the Mason's do, and so, the understanding is not straightforward. Considering, the aim of having a clear phenomena description to allow a simplification on the synthesis process implementation, the Mason's approach was chosen.

4.2. Signal flow graph - mixed representation

In the previous section, the fundamental concepts and properties associated to the SFGs were analyzed. The use of these techniques is specially suitable for the analysis of analog circuits. However, the representation of mixed analog-digital systems, as the data converters, requires an extension of the concepts and properties previously discussed. So, a new description is now mandatory to allow, on one hand, the coexistence of analog and digital signals in the graph and, on the other hand, a clear distinction between the branches where each signal class flow, in order to achieve the generation of two partitions and, therefore, the use of separate synthesis methods to each other. In the next subsections, the developed SFG description, is presented.

4.2.1. Definitions and concepts

In the new domain, analog and digital signals, the basic element for the SFG representation, the branch, must be redefined. In the classic approach, there is only one type of signal, and so, only one type of branch. However, now, two types of signal appear and simultaneously the need to generate signal partitions, so, three classes of branches were defined to cover the new signal flow alternatives, respectively, analog, digital and hybrid branches, as described on **Table 6**.

Table 6: Classes of Defined Branches for the Modified Signal Flow Graph Representation.

Class	Analog Branches	Digital Branches	Hybrid Branches
DESCRIPTION	$SA \xrightarrow{CA} SA$	$SD \xrightarrow{CD} SD$	$SA/SD \xrightarrow{CA/CD} SA/SD$
	SA – Analog Signal CA – Analog Coefficient	SD – Digital Signal CD – Digital Coefficient	

The analog branches class is composed by the branches through which an analog signal flows and where the coefficient is real, and so, it may result in an amplification, attenuation or maintenance of the signal, as the classic approach. The digital branches class is composed by the branches through which a digital signal flows and where the coefficient is of logic type. The inclusion of a coefficient of the logic type in a branch introduces a new concept on this type of representation. This concept consists on the dynamic changing of the graph structure as a function of this coefficients, this means, when they assume the logic value '1' the signal flow is allowed and when they assume the logic value '0' the signal flow is blocked. As an example, the graphs corresponding to the basic logic functions (AND, OR, NOT), as well as, the resulting subgraphs due to dynamic changes on the coefficients or logic conditions, are presented on **Table 7**.

Table 7: Basic Logic Functions Described with Digital Branches.

Condition	Graph "and"	Graph "or"	Graph "not"
$d_2 \ d_1$			
0 0			
0 1			
1 0			
1 1			

The class of hybrid branches is composed by branches where analog and digital data are presented simultaneously, and so, consists on the interface between the other two classes, this means, the interface between analog and digital signals, as included in the algorithmic data converter descriptions. However, only some combinations of analog and digital data in the branch make sense in this approach. In the case of a D/A interface, it is represented by a branch with an analog signal at the input and output, and a coefficient representing a logic condition defined by digital signals, e.g., x_1 equal x_2 . In the case of a A/D interface, it is represented by a branch with an analog coefficient, an analog signal at the input and a digital signal at the output. Here, the output node, although digital, represents the value of an analog condition, e.g., x_1 greater than x_2 . Moreover, this node results naturally, from the convergence of several hybrid branches. The logic conditions in the hybrid branches are interpreted in the same way as the digital conditions described above. In Fig. 6, a graph composed, simultaneously, by analog and hybrid branches is presented, in order to illustrate the dynamic changes as function of the digital conditions on hybrid branches. In this example, the analog signals v_{ref} , v_{gnd} e v_{x5} , the analog coefficients k_i and the digital conditions b_i were considered.

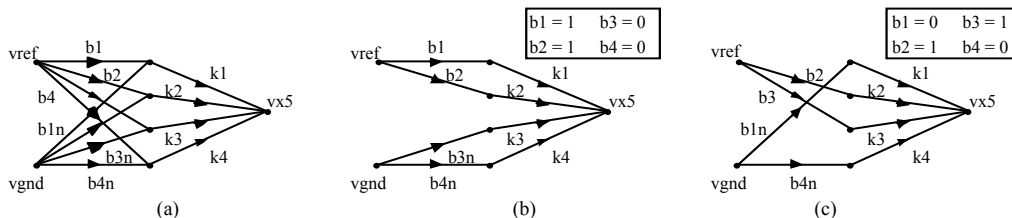


Figure 6: Analog signal flow graph with logic conditions: (a) Non instantiated structure and representation of the functions: (b) $v_{x5} = (k_1+k_2)*v_{ref} + (k_3+k_4)*v_{gnd}$ and (c) $v_{x5} = (k_2+k_3)*v_{ref} + (k_1+k_4)*v_{gnd}$.

4.2.2. SFGs simplifying rules

The achieved graph from the direct transformation of the algorithm description is not minimized due to the translation mechanisms and, also, to description redundancies. So, it is mandatory to simplify the graph, which is obtained by applying a simplification process. The simplifying rules, as exemplified on **Table 8**, are based on the generation of equivalent subgraphs with a lower number of branches and were derived based on the properties of the mixed signal SFGs.

The **rule 1** refers to the elimination of the analog branches with a null entry, *i.e.*, the branches which reproduce the action of the product absorbing element and the output behaves as the sum neutral element. The **rule 2** refers to the elimination of branches with unitary coefficient, and so, the output and input signals are equal. The **rule 3** refers to the serial association of two complementary branches. The **rule 4** refers to the parallel association of two branches and its replacement by a branch with unitary coefficient, which is removed or not depending on adjacent branches. The **rules 5 and 6** refer to the parallel and serial association of branches with real coefficients. The **rules 7 and 8** refer to the application of the left or right factorization property, depending if the common node of branches having identical coefficients is the output or the input.

Table 8: SFG Simplifying Rules.

Rule	Activation Structure	Resulting Structure
1) Absorbing Element		
2) Neutral Element		
3) Complementary I		
4) Complementary II		
5) Associative I		
6) Associative II		
7) Factorization I		
8) Factorization II		

In **Fig. 7**, the simplification process is exemplified through the transformation of an original graph structure, representing a sequence of conditional attributions into a compacted graph structure, preserving the canonical form of representation (input-condition-coefficient-...-condition-coefficient-output). The application of simplifying rules does not follow a pre-established order, though, allows the multi-solution search.

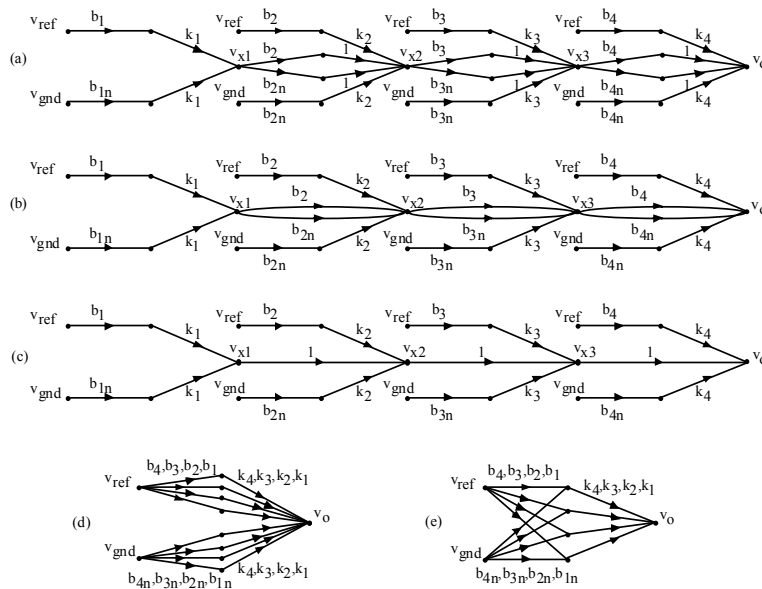


Figure 7: Simplification of a graph originated by a succession of conditional attributions, (a) Original graph, (b) after applying rule 2, (c) after applying rule 4, (d) after applying rule 2 and (e) after applying rule 7.

4.2.3. SFGs partitions

The need to decompose the synthesis process in two subprocesses, one for the analog part and other for the digital part, implies the division of the graph corresponding to a data conversion algorithm into two partitions. Based on the branch classes, previously described, the following partitions were defined:

- (1) **ANALOG PARTITION:** The set of all branches with an analog signal flowing in, which is the same of saying, all analog branches and the hybrid branches with an analog signal flowing in.
- (2) **DIGITAL PARTITION:** The set of all branches with a digital signal flowing in, which is the same of saying, all digital branches and the hybrid branches with an digital signal flowing in.

In conclusion, a set of new SFG concepts were here introduced allowing the processing of mixed signal descriptions and specially data conversion algorithms descriptions.

5. Algorithm Description to SFG Mapping

Once established the principles of signal flow through mixed analog-digital SFGs, follows the translation of the VERILOG-AMS algorithm description into a SFG, having in mind the achievement of an equivalent representation in a canonical form. The translation is performed into three phases, the first consists in a transformation of the algorithm description into an expanded graph, the second corresponds to the graph simplification and the third consists on the graph partitioning.

5.1. Translation of the algorithm description into an expanded SFG

In the implemented approach, the translation of the functional description, representing the algorithm, into a SFG is performed primitive by primitive, in a sequential manner, and so each primitive originates a subgraph of the mixed signal type. The generation of each of these subgraphs is executed based on a set of interpretation rules applied to the functional description primitives. These rules were

derived from the SFG properties presented in the previous sections. In the functional description, as analyzed in the previous section, there are two different types of functions implemented by the primitives, respectively, the attributions and the conditions. Next, the representation, in terms of SFG, for each of these functions separately and for the cases where they appear together is described in **Table 9**.

Finally, in the case of a series of conditions affecting attributions, e.g. *if-if-...* or *when-if-else...*, etc., the attributions are preceded by a unique branch corresponding to the conjunction of those conditions.

Table 9: Attributions Described by SFGs.

Operation ID	Syntax	Signal Flow Graph
Analog Attribution	$ax < + \sum_i^n k_i a_i$	
Digital Attribution	$dx = f(d_{1...n})$	
Analog Condition	<i>IF f_{a1} (>, <) f_{a2} BEGIN Region l_d END ELSE BEGIN Region l_{dn} END</i>	
Digital Condition	<i>IF f_{d1} === f_{d2} BEGIN Region l_d END ELSE BEGIN Region l_{dn} END</i>	
Conditioned Analog Attribution	<i>IF l_d</i> $ax < + \sum_i^n k_i a_i$	
Conditioned Digital Attribution	<i>IF l_d</i> $dx = f(d_{1...n})$	

5.2. Example of a SFG generation

The SFG generation is exemplified for an algorithm description of multi-step conversion. In this case, a two-step conversion with two bits per step was considered, once it allows the generation of subgraphs covering a large part of the modified SFG structures, without generating a too complex graph.

In **Fig. 8**, the algorithm description is presented with some of the subgraphs generated during the translation process, this illustrates the variety of graph structures involved. Thus, the translation starts with a subgraph corresponding to a digital condition, which allows the differentiation between two conversion states. The first conversion state starts with the sampling of the input signal, described by the attribution of the input value to a real variable. This description originates a subgraph composed by an analog branch with unitary coefficient, representing the attribution, and a hybrid branch due to the digital condition affecting the attribution. In the sequence of the

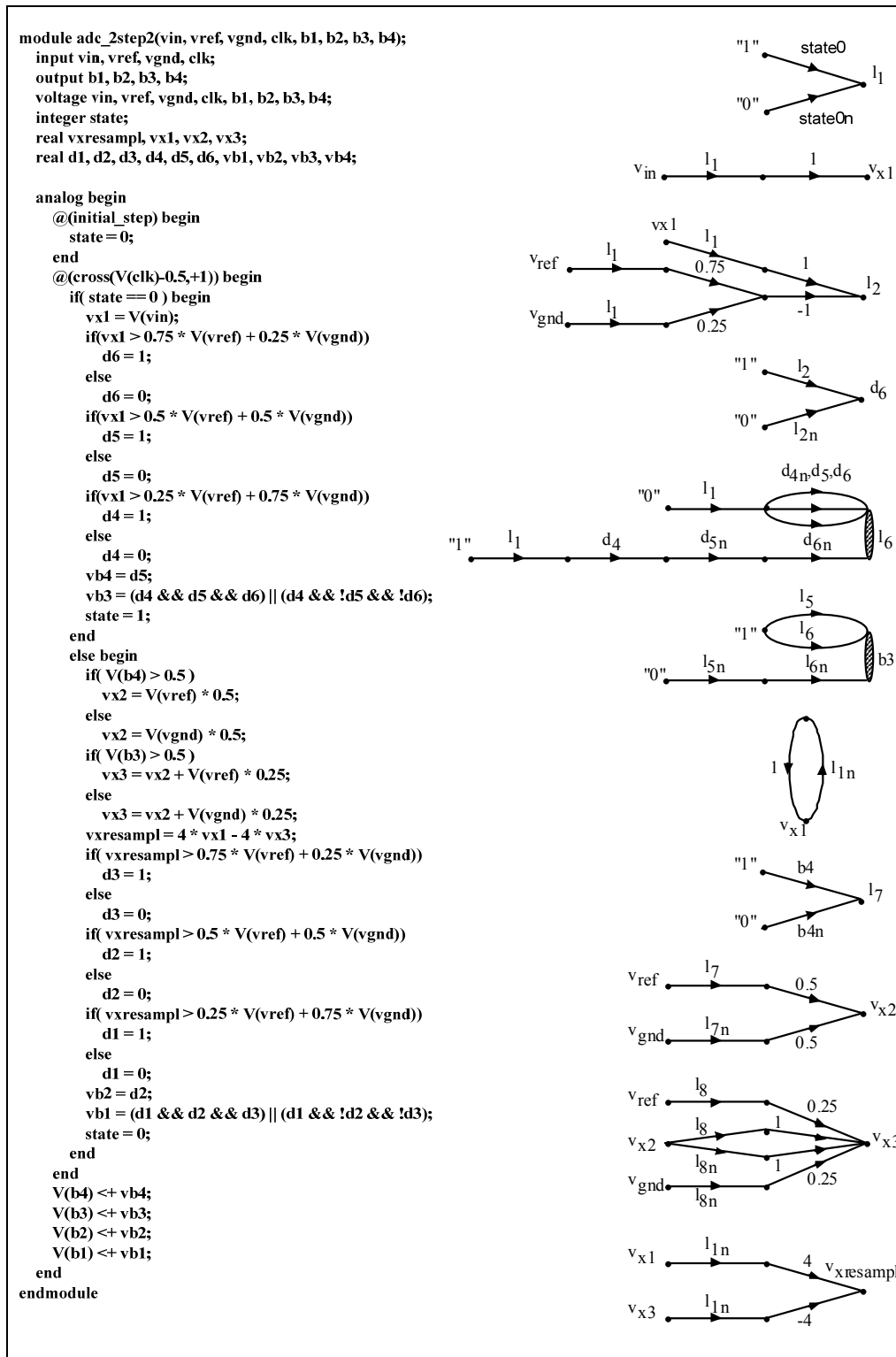


Figure 8: Example of the graph generation based on an algorithm description.

sampling, appears the first condition described based on analog signals, an analog condition, which still depends on the previous digital condition. The index associated to the conditions is determined based on the identification of different conditional regions. The next subgraph refers to a digital attribution where is outlined the

identification of different conditional regions. The four subgraphs described above illustrate the fundamental types of attributions and conditions in this kind of descriptions, however, the generation of other subgraphs are presented to allow the comprehension of the final structure.

The next subgraphs refer to the generation of bit b_3 resulting from the transformation of the thermometer scale, and represented by the signals d_6 , d_5 and d_4 , into natural binary code. The second conversion state starts with the description of the input hold, which is performed if the condition l_{1n} is true, and so, the inclusion of an hybrid branch to show this dependency. Next, a digital condition illustrates the effect of the results achieved on the first conversion step. The following structure presents a typical case of a D/A interface description, where an analog signal value, v_{x2} , depends on digital conditions, $l_7 \in l_{7n}$. Then, another D/A structure is illustrated, having a common node with the previous one. Finally, the subgraph corresponding to the generation of the residual signal to be converted in the second phase is presented.

The complete graph associated to the algorithm description results from the interconnection of all determined subgraphs during the interpretation process. The analog partition is presented in **Fig. 9(a)**, where the previous determined subgraphs can be observed. The complete graph is here presented, for simplicity, with less redundant branches than is actually generated, as can be verified in **Table 10**. In **Fig. 9(b)**, the final graph obtained after the simplifying process is shown, this constitutes the base for the functional topology generation to be discussed in the next section.

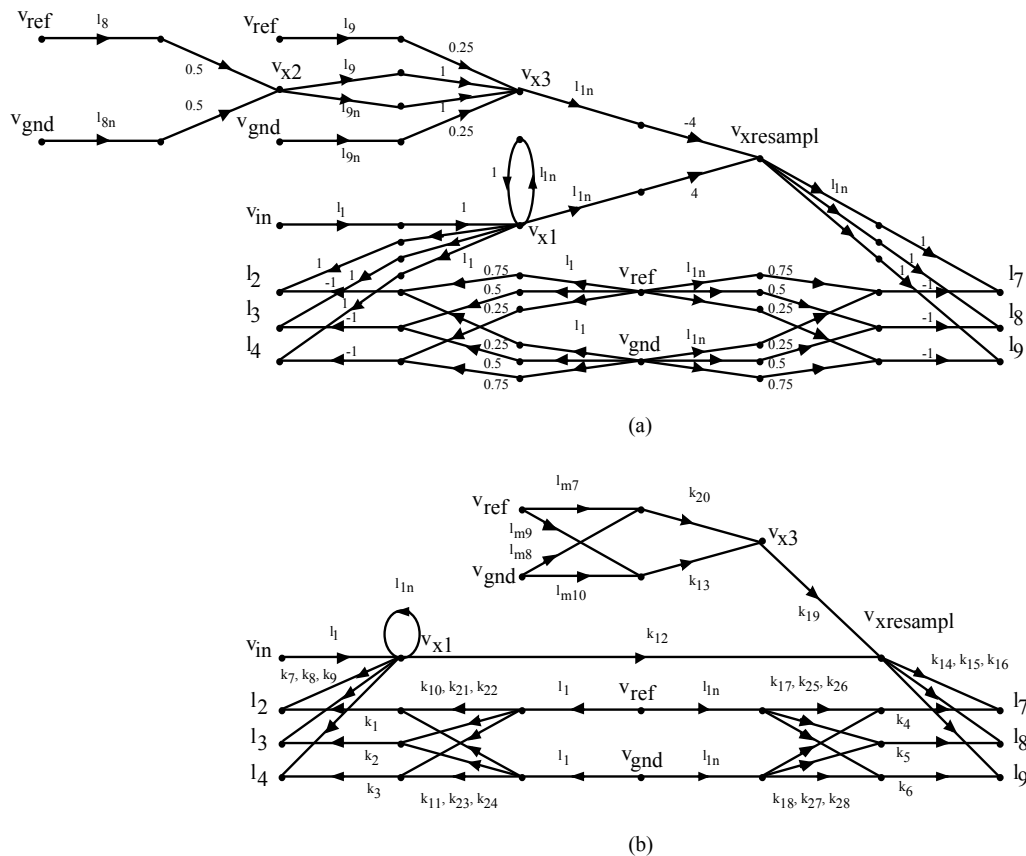


Figure 9: Analog partition from a SFG achieved for a two-step data conversion algorithm description, respectively, (a) before and (b) after simplification.

5.3. Performance analysis for the algorithm to SFG mapping

The generation and simplification of the SFG corresponds to the first of the three phases implemented by the synthesis process. The performance analysis of this phase, is made using three common algorithms and a fourth obtained by a mixed of the first two. So, on one hand, the parallel, the successive approximation and the two-step algorithm were considered, on the other hand, a successive approximation algorithm with m bits per step, was used. In **Table 10**, the results obtained are presented, particularly, the original and simplified graph size.

The achieved results show that, although, in the more adverse case the SFG is too large to be analyzed by hand, its size does not prevent the automatic processing. More, it shows the capability to process new algorithms, which means an enlargement of the design domain when compared to the traditional synthesis tools for this kind of systems.

In **Table 11**, the SFGs achieved for the referred algorithms, are presented. In all these cases, the identification of the coefficient values and the location of the hybrid branches, allows the observation of some isolated functionalities that will be automatically recognized during the functional topology generation, as will be described in the next section.

Table 10: Performance Analysis for SFGs Generation and Simplification.

Algorithm Characterization		SFG Generation		SFG after Simplification		Algorithm Characterization		SFG Generation		SFG after Simplification	
Type	N. Bits	No. Branches	No. Branches	Type	N. Bits	No. Branches	No. Branches	Type	N. Bits	No. Branches	No. Branches
Successive Approximation	4	56	16	Two-Step	2+2	96	38		2+2	96	38
	8	112	28		4+4	388	140		4+4	388	140
	12	168	40		5+5	754	271		5+5	754	271
	16	224	52		6+6	1472	530		6+6	1472	530
Parallel	2	37	16	S.A. and Parallel	2+2	127	38		2+2	127	38
	4	169	64		2+2+2+2	239	62		2+2+2+2	239	62
	6	697	256		4+4	347	110		4+4	347	110
	8	2809	1024		4+4+4+4	571	158		4+4+4+4	571	158

Table 11: SFGs for Different Conversion Algorithms.

Algorithm	SFG Simplified
Succe. Approx. 4 bit: $k_1=0.0625; k_2=1;$ $k_3=-1; k_4=0.125;$ $k_5=0.25; k_6=0.5.$	
Parallel 2 bit: $k_1=k_2=k_3=-1;$ $k_4=k_5=k_6=1;$ $k_7=k_12=0.75;$ $k_8=k_10=0.25;$ $k_9=k_11=0.5.$	

Table 11: SFGs for Different Conversion Algorithms (cont).

Algorithm	SFG Simplified
Two-Step - 2+2 bit: $k_1=k_2=k_3=k_4=k_5=k_6=-1$; $k_7=k_8=k_9=k_{14}=k_{15}=-k_{16}=1$; $k_{10}=k_{17}=k_{24}=-k_{28}=0.75$; $k_{11}=k_{13}=-k_{18}=k_{22}=k_{26}=0.25$; $k_{12}=4$; $k_{19}=-4$; $k_{20}=-k_{21}=k_{23}=k_{25}=k_{27}=0.5$	
S.A. & Parallel 2+2 bit: $k_1=k_2=k_3=-1$; $k_4=k_5=0.065$; $k_6=k_9=k_{12}=1$; $k_7=k_{14}=0.75$; $k_8=k_{13}=k_{16}=k_{19}=0.25$; $k_{10}=k_{11}=k_{17}=k_{20}=1$; $k_{15}=k_{18}=0.125$.	

6. SFG to Topology Mapping

6.1. Topology generation - analog partition

The topology consists of a set of functional blocks connected in order to describe the conversion algorithm. Therefore, the topology generation corresponds to the identification of those blocks in the modified SFG description associated to the algorithm description. In the following subsections, the developed methodology for the topology generation is discussed.

6.1.1. Methodology definition

The selected functional blocks identification method is based on symbolic processing. This choice was taken since the objective is to identify functional structures, whose configuration may change, during the algorithm execution and according to the coefficients values. Thus, it is possible to identify similar structures even if parameterized with different values, which will be needed only during the sizing task.

In order to implement the functional block identification process, two main alternatives occur, on one hand, the symbolic expressions recognition and, on the other hand, the pattern recognition. The first choice consists of employing symbolic techniques to extract an expression from the graph description, this is, usually, used when the goal is to determine the relation between the signals in two well defined nodes of a circuit. However, in the present case, the application of this technique is not limited to the generation of an expression, which relates just two signals in a circuit, here, a set of expressions must be generated, for all nodes combinations, in order to perform the comparison with the expressions of each functional block. The existence of blocks with multiple outputs and/or inputs and the large complexity of graphs lead to a combinatory explosion in the search, with obvious problems in terms of time consumption, thus, the first choice was abandoned. The second choice

consists of applying pattern recognition techniques to the identification of subgraphs associated to functional blocks, therefore, eliminating the need to generate expressions, once, it just has to analyze the type of branch and its connectivity. In this case, the complexity of the search process is reduced but the functional blocks patterns, in terms of SFGs, must be included in the library, which is done only once. For these reasons, the pattern recognition approach was implemented following the methodology described by the flowchart of **Fig. 10**.

The methodology is composed by three main phases, respectively, the selection of the functional block from the library, the pattern search for the selected block and graph reduction after the pattern recognition. The first phase implies the development of a functional block library with the corresponding SFG description for each block. So, the quantity and quality of functional blocks described in library will condition the topology generation process. Next, the pattern search for the selected functional block is executed. Finally, in the case of success on the previous step, the space search must be reduced, *i.e.*, the graph reduced, to simplify the search process for the next functional blocks. A detailed discussion of these phases is presented in the next sections.

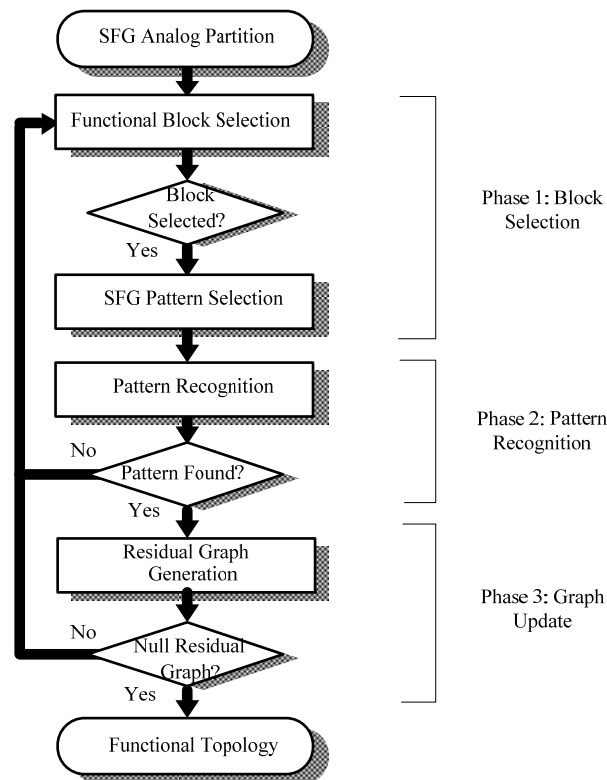


Figure 10: Methodology for the functional topology generation.

6.1.2. *Functional block library*

The phase corresponding to the functional block selection is based on a functional block library. The design of the library starts by identifying the functional blocks needed and, usually, used in the data conversion topologies. The description of these functional blocks is done in a generic base, as exemplified on **Table 12**. Then, together with the generic description of the functional block, each library entry must

include the SFG corresponding to the functional block, *i.e.*, its pattern, to support the search on the algorithm graph. The pattern for each block is determined through the translation of the block function in terms of SFG, according to the rules described previously. Therefore, each entry of the library is composed by both the functional blocks and the corresponding graph described generically.

During the topology generation process the functional blocks are selected for the search task in a sequential way. The selection is restarted, by the first element, always that a block is identified on the graph, and fails when it arrives to the end of the library list without occurring any pattern identification.

In the implemented system, the functional blocks library behaves as an open module allowing its permanent updating.

Table 12: Functional Blocks and the Corresponding Pattern in Terms of SFGs.

Functional Block Description	Functional Block	Graph Pattern
Sampling and Hold- Analog output, v_{out} , samples the analog input, v_{in} , in the time instants associated by the condition $l1$ and holds the signal in the other instants.		
Multiple Adder (not Amplified; not inverted) - Analog output signal corresponding to the sum of the input signals, v_{in_i} , weighted by the real coefficients $k_i \in]0,1[$.		
Multiple Adder Digitally Controlled (not Amplified; not inverted) - Analog output signal corresponding to the sum of the input signals, v_{ref_1} and v_{ref_2} , weighted by the coefficients $k_i \in]0,1[$ and $b_i \in \{0,1\}$.		
Single Reference Multiple Divider (Digitally Controlled) - Analog output signals corresponding to the entry weighted by the coefficients $k_i \in]0,1[$.		
Double Reference Multiple Divider (Digitally Controlled) - Analog output signals corresponding to the imputes sum weighted by the real coefficients $k_i \in]0,1[$ and with $k_{1i} + k_{2i} = 1$.		
Difference Amplifier (Digitally Controlled) - Analog output signal corresponding to the input difference amplified by the factor $k_i > 1$.		
Multiple Comparator (Digitally Controlled) - Digital Outputs resulting from the comparison between two analog signals weighted by the real coefficients $k_i \in]0,1[$.		

6.1.3. Functional block identification in the SFG

The functional block identification consists of the search of the corresponding subgraph on the graph describing the algorithm. The pattern recognition process is performed by a sequence of steps, and not only one step, due to the generic description in the library. Thus, although the identification process is more complex,

the number of library elements is lower, avoiding, on one hand, the need to describe the same block for different dimensions and, on the other hand, the use of basic structures originating topologies with a higher number of blocks. In both cases the number of iterations would have been obviously larger. The sequence of pattern recognition steps, consists of, first, the identification of particular pattern substructures and, then, in the identification of the substructures with variable dimensions. In order to minimize the number of blocks composing the topology the search is oriented to first select the larger blocks. In **Fig. 11**, the process of recognizing the pattern of a functional block on the graph of the two-step conversion algorithm is described. In this case, the searched pattern corresponds to the functional block labeled as multiple comparator. The first step, refers to a search of a particular substructure, a simple comparison which is composed by two analog branches with analog coefficients and converging both, branches, in a digital node. On the presented graph there are six structures with the referred characteristics, e.g. (k_4, k_{14}) , (k_5, k_{15}) , (k_6, k_{16}) , (k_1, k_7) , (k_2, k_8) and (k_3, k_9) , and without having any special reason to prefer one of them, the option is for the first pair in the solution list. The second step, consists of determining the maximum dimension for the functional block being identified. That is obtained by searching for other branches with the same characteristics of the previous ones and, that also, have a node in common between them and the first pair. In the present case, the pairs (k_5, k_{15}) and (k_6, k_{16}) verify the desired characteristics, and so, the functional block has a maximum dimension of three structures. The identification of the three blocks of simple comparison or one block of multiple comparison is equally correct in functional terms, however, leads to more compact topologies and a faster process associated to the topology generation by reducing the numbers of iterations. Finally, on third step the existence of digital conditions affecting the block is verified.

The multiple solution search through the block selection by different orders may allow the generation of different topologies for the same algorithm, but it also leads to the appearance of similar solutions achieved by different selection orders, which is undesirable. For this reason, after the identification of each functional block, the set of partial solutions are compared, in order to eliminate the equivalent ones, as illustrated in **Fig. 12**. Finally, the implemented system allows the activation or not of the multiple solution generation process.

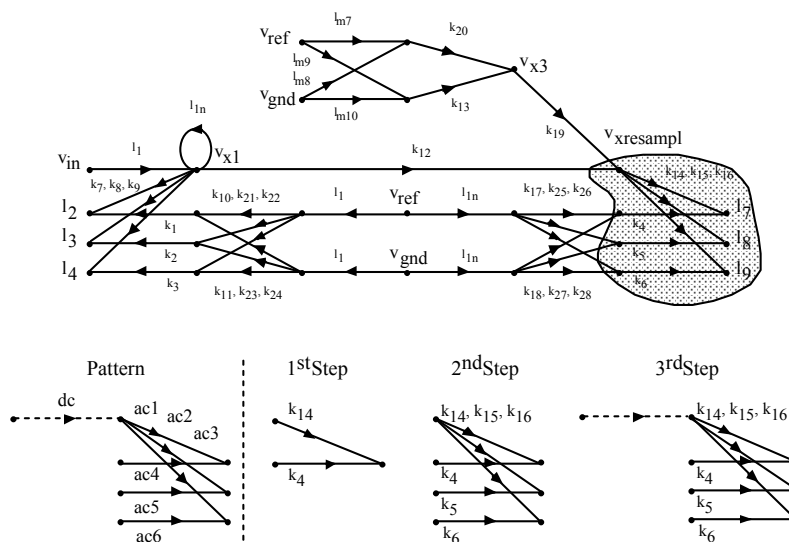


Figure 11: Pattern identification process.

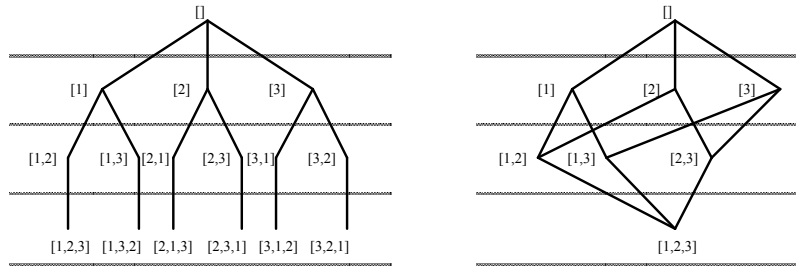


Figure 12: Effect of eliminating the redundant solutions.

6.1.4. Residual graph generation

The recognition of a pattern on the algorithm graph adds a functional block to the topology and, consequently, obligates to a reduction on the search space, the graph, as illustrated in Fig. 13, to allow further functional blocks identification.

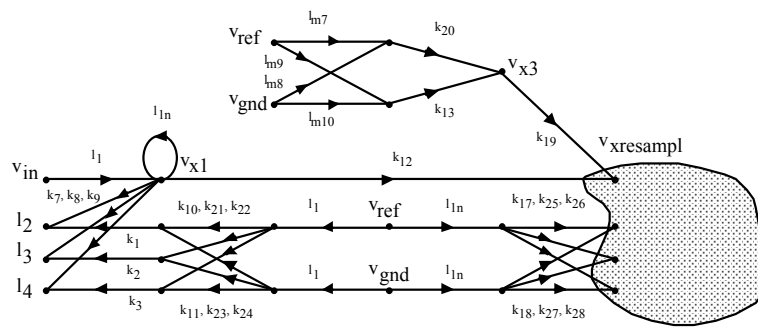


Figure 13: Residual graph determination by removing the identified pattern.

However, the residual graph is not obtained just by removing the branches corresponding to the identified pattern from the graph. The elimination of the branches must consider the existence of other branches belonging to the graph describing the algorithm and not to the pattern, but converging on it, and so affecting the forward part of the pattern, or diverging from it, and so depending from the backward part of the graph. Therefore, the pattern internal nodes where some other branches converge or diverge are, here, referred as by critical nodes. In Fig. 14(a) the reduction process relative to a pattern having a critical node of convergence is illustrated. In this case, in order to preserve the context, while removing the pattern, a copy of the forward part of the pattern must be added, as well as, a set of additional nodes and branches to generate sum points. In Fig. 14(b) the reduction process relative to a pattern having a critical node of divergence is illustrated. In this case, the context is preserved by adding a copy of the backward part of the graph.

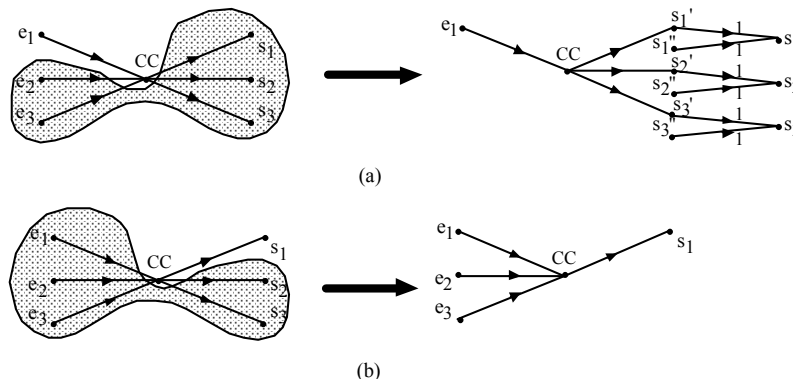


Figure 14: Residual graph generation with critical nodes of (a) convergence and (b) divergence.

In the working example, illustrated in Fig. 11, if the functional block identification was performed with priority to blocks of the smallest dimension, the identified block would have been a single comparator, e.g. (k4, k14), with a critical node of divergence. In this case, the critical node of divergence would have implied maintenance of the backward part of the graph, as illustrated in Fig. 15.

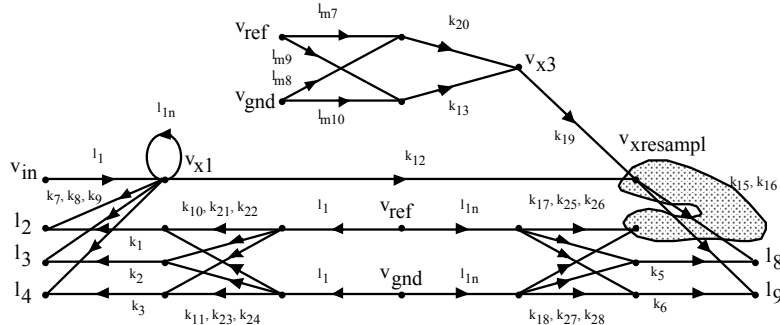


Figure 15: Residual graph generation in the case of a critical node of divergence.

Therefore, to deal with the graph reduction problem, in a generic way, all the possible combinations must be analyzed. On Table 13, the different alternatives are summarized, as well as, the corresponding transformations.

Table 13: Graph Reduction Alternatives.

Critical Node Type:	Graph Reduction Type:
none	complete elimination of the identified pattern
only convergence	elimination of the pattern backward part and introduction of sum structures
only divergence	elimination of the pattern forward part
divergence node prior to the convergence nodes	elimination of the branches between the critical nodes and introduction of sum structures
convergence node prior to the divergence nodes	no reduction

The presented analysis may, now, be synthesized with following set of graph reduction rules:

- **Rule 1:** in the case of various critical nodes of convergence, the forward substructure of the pattern to the first critical nodes is maintained and sum structures are introduced.
- **Rule 2:** in the case of various critical nodes of divergence, the backward substructure of the pattern to the last critical nodes is maintained.
- **Rule 3:** in the case of the simultaneous existence of critical nodes of convergence and divergence, there is a reduction, only, if the most distant critical node of divergence, from the input, is closer than the nearest critical node of convergence.

6.1.5. Example of a functional topology generation

As an example of topology generation, for the analog partition, the two-step conversion algorithm was selected. On Table 14, the successive iterations leading to the functional topology are described. In each iteration, first, the most complex functional block not yet considered is selected, then, the SFG pattern for the block is determined, after, the pattern is searched in the graph structure, finally, the graph is reduced and the process is repeated until no residual graph exists, i.e., the topology is completely generated. The present example, is initiated with the selection of the highest priority block, i.e. the reference divider block. The existence of two subgraphs

Table 14: Functional topology generation

Methodology	Action	Steps 1 and 2	Steps 3 and 4
	<p>S E L E C T I O N</p>		
<p>I D E N T I F I C A T I O N</p>			
<p>R E D U C T I O N</p>	<p>F U N C T I O N A L T</p>		
<p>F U N C T I O N A L T</p>			

Table 14: Functional topology generation (cont.)

Step 5	Step 6	Step 7

corresponding to this type of block, allow, immediately, the identification, on the first two steps, of two functional blocks belonging to the topology. Afterwards, the block with higher priority from the remaining ones is selected, *i.e.* the multiple comparator block. Alike the previous case, two other occurrences of the selected block are again detected and, so, in the third and fourth steps another two blocks are identified.

Then, at the end of the fourth step the topology being generated has already four blocks and their connections defined according to the algorithm description represented by SFGs. In the last three steps other three blocks are identified, namely, the multiple adder, the difference amplifier and the sample and hold blocks, allowing, this way, the complete graph reduction and, consequently, topology generation.

The achieved topology, in this example, allows the verification of the perfect matching between the fully automatically generated topology and the typical topology for this converter, which proofs the applicability of the developed methodology.

Table 15: Functional Topologies Generated for Different Conversion Algorithms.

Algorithm	Functional Topology
Suc. Appro. 4 bit	
Parallel 2 bit	
Two-Step 2+2 bit	
Suc. Appro. & Parallel 2+2 bit	

6.1.6. Performance analysis for topology generation

The topologies generated based on the SFG descriptions are presented in **Table 15**. In these descriptions, there is, always, a set of parameters associated to each block.

These parameters consist both on the digital conditions and on the analog coefficients, which will constrain the generation of the electrical topology and the sub-block specifications, which is out of scope for this chapter.

The test results, in **Table 16**, were obtained for several data converter resolutions showing that although the graph increases the number of blocks remain constant on each case. This means that despite the increase in resolution the number of identifying steps is identical and, therefore, the process of generating the topology is easily scalable.

6.2. Topology validation - simulation

The correct topology simulation validates the synthesis process by confirming the suitability of the selected functional blocks and their interconnections. In the next sections the selected functional blocks are described, in VERILOG-AMS, as well as, the way the simulation is automatically prepared for the CADENCE[®] IC Design Environment.

Table 16: Effect of the Functional Selection Order in the Synthesis.

Algorithm/Graph Characterization			Topology Generation
Algorithm	Resolution	No. Branches	No. Blocks
Successive Approximation	4	16	3
	8	28	3
	12	40	3
	16	52	3
Parallel	2	16	3
	4	64	3
	6	256	3
	8	1024	3
Two-Step	2+2	38	7
	4+4	140	7
	5+5	271	7
	6+6	530	7
Succ. Appro. & Parallel	2+2	38	5
	2+2+2+2	62	5
	4+4	110	5
	4+4+4+4	158	5

6.2.1. Functional blocks modeling

The topology validation by functional simulation implies the development of models to represent all the blocks described in the library. Thus, maintaining the philosophy of one tool one simulator. The models whose general structure is described on **Fig. 16**, are characterized by the description of mathematical relations between inputs and outputs.

```

module modelname( p1, ..., pn);
  input|output|inout pin p1, ..., pn;
  electrical|voltage|current pin p1, ..., pn;
  parameter real k1 = val_k1;
  ...
  parameter real kn = val_kn;
  real s1, ..., sn;
  analog begin
    @(initial_step) begin
      (* initialization *)
    end
    if (analysis("dc"|"ac"|"tran"|"noise")) begin
      (* attributions *)
    end
    else begin
      (* attributions *)
    end
    (* attributions *)
  end
endmodule

```

Figure 16: Description structure for a functional block.

In the implemented system the models for the various functional blocks are generated automatically, with the sizing and connections determined during the topology generation, to allow their immediate use by the simulator. Next, some of the models generated for the last functional topology, illustrated in **Table 15**, are presented.

In this example, a low resolution was intentionally chosen to simplify the analysis, however, the discussion is also valid for other resolution values. The analysis of the topology allows the identification of four distinct analog blocks, namely, the sample and hold, the double reference multiple divider, the multiple comparator and the multiple adder digitally controlled. On **Fig. 17**, the sample and hold block is described. By a simple analysis the equivalence between this model and the graph presented on **Table 12**, on one hand, the interface between the model and outside world is made with four signals, respectively, the output, the input and the two control conditions, on the other hand, the body of the model describes the relation between output and input as a function of the conditions.

```

module snh_fm(out, in, ind1, ind1n);
  input in, ind1, ind1n;
  output out;
  electrical out, in, ind1, ind1n;
  real last_value;

  analog begin
    if (V(ind1) > 2.5 )
      last_value = V(in);
    V(out) <+ last_value;
  end
endmodule

```

Figure 17: S&H functional block description.

```

module mdvr4_fm(out1, out2, out3, vref, vgnnd);
  input vref, vgnnd;
  output out1, out2, out3;
  electrical out1, out2, out3, vref, vgnnd;
  parameter real k1 = 2.5e-1, k2 = 7.5e-1, k3 = 5e-1,
    k4 = 5e-1, k5 = 7.5e-1, k6 = 2.5e-1;

  analog begin
    V(out1) <+ k1 * V(vref) + k2 * V(vgnnd);
    V(out2) <+ k3 * V(vref) + k4 * V(vgnnd);
    V(out3) <+ k5 * V(vref) + k6 * V(vgnnd);
  end
endmodule

```

Figure 18: Double reference multiple divider functional block description.

On **Fig. 18**, the double reference multiple divider is described, in this case, and in the remaining ones, the description is dependent on the specified resolution. This is outlined by the interface elements, once the number of outputs depends on the resolution, and the entries are always two, corresponding to the upper and lower limit of the outputs. The relation between inputs and outputs, *i.e.* the analog coefficients of the corresponding graph, is expressed by the real parameters associated to the model and the attributions made on the model body.

On **Fig. 19**, the multiple comparator block is described, with the three outputs corresponding to the two bit codification in each conversion step. In this case, the choice was for an output between 0 and 5 volt, but, other values may be specified through the synthesis system.

```

module mcmp3_fm(out1, out2, out3, inpc, inn1, inn2, inn3);
  input inpc, inn1, inn2, inn3;
  output out1, out2, out3;
  electrical out1, out2, out3, inpc, inn1, inn2, inn3;
  parameter real k1 = 1, k2 = -1, k3 = -1, k4 = -1;

  analog begin
    if ( V(inpc) * k1 + V(inn1) * k2 > 0 )
      V(out1) <+ 5;
    else
      V(out1) <+ 0;
    if ( V(inpc) * k1 + V(inn2) * k3 > 0 )
      V(out2) <+ 5;
    else
      V(out2) <+ 0;
    if ( V(inpc) * k1 + V(inn3) * k4 > 0 )
      V(out3) <+ 5;
    else
      V(out3) <+ 0;
  end
endmodule

```

Figure 19: Multiple comparator functional block description.

```

module madc4_fm(out, vref, vgn, ind1, ind2, ind3, ind4, ind1n);
  input vref, vgn, ind1, ind2, ind3, ind4, ind1n;
  output out;
  electrical out, vref, vgn, ind1, ind2, ind3, ind4, ind1n;
  parameter real k1 = 5e-1, k2 = 2.5e-1, k3 = 1.25e-1, k4 = 6.25e-2;
  real s1, s2, s3, s4, s5;

  analog begin
    if ( V(ind1) > 2.5 )
      s1 = 1;
    else
      s1 = 0;
    if ( V(ind2) > 2.5 )
      s2 = 1;
    else
      s2 = 0;
    if ( V(ind3) > 2.5 )
      s3 = 1;
    else
      s3 = 0;
    if ( V(ind4) > 2.5 )
      s4 = 1;
    else
      s4 = 0;
    if ( V(ind1n) > 2.5 )
      s5 = 1;
    else
      s5 = 0;
    V(out) <+ V(vref) * s1 * k1 + V(vref) * s2 * k2 + V(vref) * s3 * k3 + V(vref) * s4 * k4 + V(vgn) * s5;
  end
endmodule

```

Figure 20: Digitally controlled multiple adder functional block description.

The digitally controlled multiple adder block, described on **Fig. 20**, corresponds to the sum of the reference values, weighted by the analog coefficients and controlled by the digital conditions. In the generated topology, two of these structures were included corresponding, respectively, to the generation of the upper and lower voltage limits, allowing, this way, to establish the range in which the values are generated by the double reference multiple divider.

6.2.2. Topology simulation

The simulation file is generated after determining the models and requires only the netlist, which is obtained based on the generated topology. In **Fig. 21**, the corresponding schematic is illustrated. This is composed by the analog blocks identified during the SFG to topology translation task.

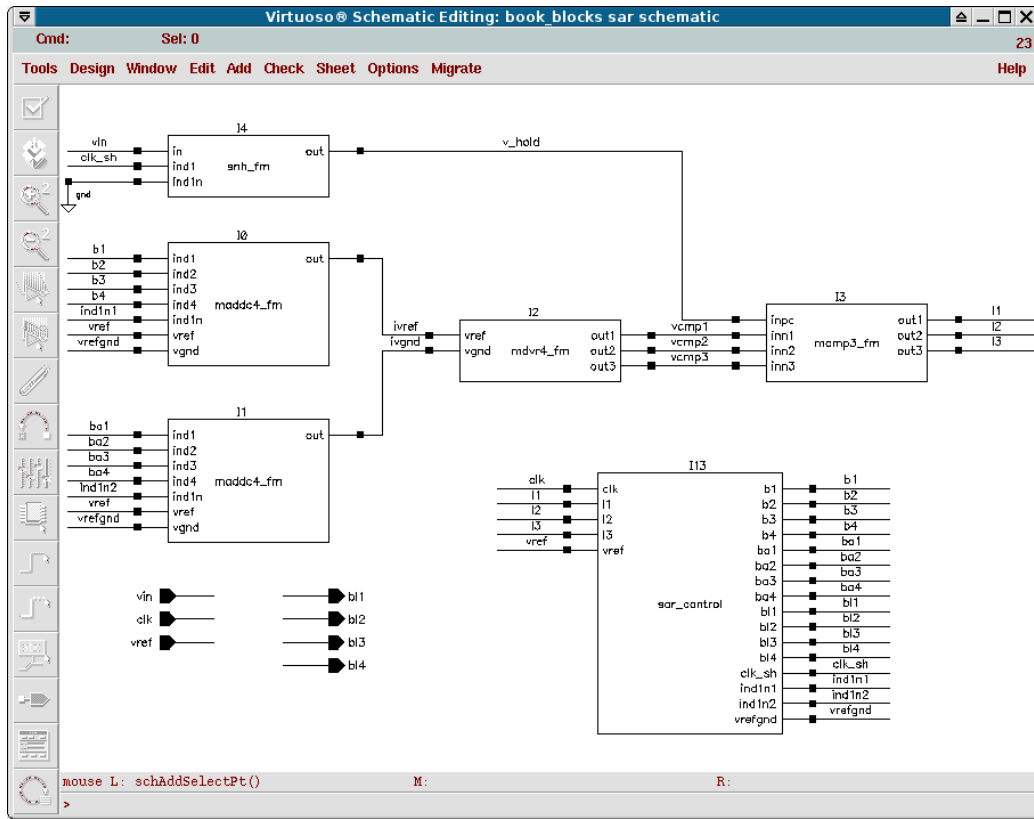


Figure 21: Spectre functional topology schematic.

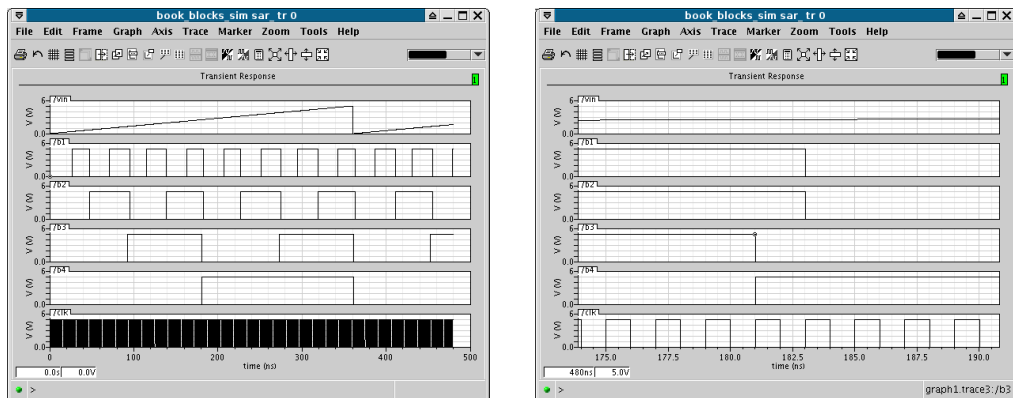


Figure 22: Spectre functional topology simulation.

The previous description allows the direct simulation and the confirmation of the synthesis correctness by comparing the achieved results with the direct algorithm simulation. In **Fig. 22**, the functional simulation results are illustrated. In this case, just some of the more relevant signals were presented, although, a larger list was specified in the netlist.

7. Conclusion

The algorithm-driven methodology developed for the synthesis and characterization of data converter systems, using extensively symbolic methods, allows the automatic exploration of a broad range of conversion algorithms without implying a previous description of the system topology as in the traditional approaches. Although still limited at lower level by a required interface with other tools, the generality of this approach opens future opportunities to other classes of mixed-signal systems. Moreover, the high-level of specifications and the flexibility of input descriptions show that the expansion to analogue filters synthesis, specified either on the frequency domain or on the time domain, is perfectly feasible with such a methodology.

References

- [1] D. Leenaerts, G. Gielen, and R. Rutenbar, "CAD solutions and outstanding challenges for mixed-signal and RF IC design", in *Proc. IEEE/ACM International Conference on Computer Aided Design*, 2001, pp. 270-277.
- [2] E. Martens and G. Gielen, "Classification of analog synthesis tools based on their architecture selection mechanisms", *Integration, the VLSI Journal*, vol. 41, pp. 238-252, February 2007.
- [3] P. E. Allen and P. R. Barton, "A Silicon compiler for successive approximation A/D and D/A converters", in *Proc. IEEE Custom Integrated Circuits Conference*, 1986, pp. 552-555.
- [4] W. J. Helms and B. E. Byrket, "Compiler generation of A to D converters", in *Proc. IEEE Custom Integrated Circuits Conference*, 1987, pp. 161-164.
- [5] J. G. Kenney and L. R. Carley, "CLANS: a high-level synthesis tool for high resolution data converters", in *Proc. IEEE International Conference on Computer Aided Design*, 1988, pp.496-499.
- [6] G. Jusuf, P. R. Gray, and A. L.Sangiovanni-Vincentelli, "CADICS - cyclic analog-to-digital converter synthesis", in *Proc. IEEE International Conference on Computer Aided Design*, 1990, pp. 286-289.
- [7] S. G. Sabiro, P.Senn, and M. S. Tawfik, "HiFADiCC: a prototype framework of a highly flexible analog to digital converters silicon compiler", in *Proc. IEEE International Symposium on Circuits and Systems*, 1990, pp.1114-1117.
- [8] N. C. Horta, J. E. Franca, and C. A. Leme, "Framework for architecture synthesis of data conversion systems employing binary-weighted capacitor arrays", in *Proc. IEEE International Symposium on Circuits and Systems*, 1991, pp.1789-1792.
- [9] N. C. Horta, J. Vital, and J. E. Franca, "Automatic multi-level macromodel generation for data conversion systems employing binary-weighted capacitor arrays", in *Proc. IEEE International Symposium on Circuits and Systems*, 1992, pp. 2561-2564.
- [10] J. C. Vital, N. C. Horta, N. S. Silva, and J. E.Franca, "CATALYST: a highly flexible CAD tool for architecture-level design and analysis of data converters", in *Proc. IEEE European Design Automation Conference*, 1993, pp. 472-477.
- [11] E. S. Ochotta, R. A.Rutenbar, and L. R. Carley, "ASTRX/OBLX: tools for rapid synthesis of high-performance analog circuits", in *Proc. 31th ACM/IEEE Design Automation Conference*, 1994, pp. 24-30.
- [12] F. Medeiro, B. Perez Verdu, A. Rodriguez Vazquez, and J.L. Huertas, "A vertically integrated tool for automated design of modulators", *IEEE Journal of Solid-State Circuits*, vol. 30, pp. 762-772, July 1995.
- [13] H. Chang, E. Liu, R. Neff, E. Felt, E. Malavasi, E.Charbon, A. Sangiovanni-Vincentelli, and P. R. Gray, "Top-down, constraint-driven methodology based generation of n-bit interpolative

- current source D/A converters”, in *Proc. IEEE Custom Integrated Circuits Conference*, 1994, pp. 369-372.
- [14] R. Phelps, M. Krasnicki, R. Rutenbar, L. R. Carley, and J. Hellums, “ANACONDA: simulation-based synthesis of analog circuits via stochastic pattern search”, *IEEE Transactions on Computer-Aided Design*, vol. 19, pp. 703-717, June 2000.
- [15] G. Gielen *et al.*, “An analog module generator for mixed analog/digital ASIC design”, *International Journal on Circuit Theory and Applications*, vol. 23, pp. 269–283, July-August 1995.
- [16] M. Hershenson, S. Boyd, and T. Lee, “GPCAD: a tool for CMOS op-amp synthesis”, in *Proc. IEEE/ACM International Conference Computer-Aided Design*, 1998, pp. 296–303.
- [17] N.C. Horta and J.E. Franca, “High-level data conversion synthesis by symbolic methods”, in *Proc. of IEEE International Symposium on Circuits and Systems*, 1996, pp. 802-805.
- [18] N. Horta, “Analogue and mixed-signal systems topologies exploration using symbolic methods”, *Analog Integrated Circuits and Signal Processing*, vol. 31, pp. 161-176, May 2002.
- [19] Glenn Wolfe and Ranga Vemuri, “Extraction and use of neural network models in automated synthesis of operational amplifiers”, *IEEE Transactions on Computer Aided Design*, vol. 22, February 2003.
- [20] M.A. El-Beltagy, P.B. Nair, and A.J. Keane, “Metamodeling techniques for evolutionary optimization of computationally expensive problems: promises and limitations”, in *Proc. of Genetic and Evolutionary Computation Conference*, 1999, pp. 196-203.
- [21] G. Gielen and R. Rutenbar, “Computer-aided design of analog and mixed-signal integrated circuits”, *Proceedings of the IEEE*, vol. 88, pp. 1825 – 1854, December 2000.
- [22] R. Sommer, I. Rugen-Herzig, E. Hennig, U. Gatti, P. Malcovati, F. Maloberti, K. Einwich, C. Clauss, P. Schwarz, and G. Noessing, “From system specification to layout: seamless top-down design methods for analog and mixed-signal applications”, in *Proc. of Design Automation and Test in Europe Conference and Exhibition*, 2002, pp. 884 – 891.
- [23] C. Toumazou and C. Makris, “Analog IC design automation: part I - automated circuit generation: new concepts and methods”, *IEEE Transactions on Computer Aided Design*, vol. 14, pp. 218–238, February 1995.
- [24] F. El-Turky and E. Perry, “BLADES: an artificial intelligence approach to analog circuit design”, *IEEE Transactions on Computer Aided Design*, vol. 8, pp. 680–692, June 1989.
- [25] M. Degrauwe *et al.*, “IDAC: an interactive design tool for analog CMOS circuits”, *IEEE Journal of Solid State Circuits*, vol. 22, pp. 1106–1115, December 1987.
- [26] R. Harjani, R. Rutenbar, and L. R. Carley, “OASYS: a framework for analog circuit synthesis”, *IEEE Transactions on Computer Aided Design*, vol. 8, pp. 1247–1265, December 1989.
- [27] C. Leme, N.C. Horta, J.E. Franca, A. Yufera, A. Rueda, J.L. Huertas, L. Paris, and T. Oses, “Flexible silicon compilation of charge redistribution data conversion systems”, in *Proc. IEEE Midwest Symposium on Circuits and Systems*, 1991, pp. 403-406.
- [28] N.C. Horta, J.E. Franca, and C.A. Leme, “Framework for architecture synthesis of data conversion systems employing binary-weighted capacitor arrays”, in *Proc. IEEE International Symposium on Circuits and Systems*, 1991, pp. 1789-1792.
- [29] H.Y. Koh, C.H. Sequin, and P.R. Gray, “OPASYN: a compiler for CMOS operational amplifiers”, *IEEE Transactions on Computer Aided Design*, vol. 9, pp. 113-125, February 1990.
- [30] J.P. Harvey, M.I. Elmasry, and B. Leung, “STAIC: an interactive framework for synthesizing CMOS and BICMOS analog circuits”, *IEEE Transactions on Computer Aided Design*, vol. 11, pp. 1402-1417, November 1992.
- [31] P.C. Maulik and L.R. Carley, “Automating analog circuit design using constrained optimization techniques”, in *Proc. IEEE/ACM International Conference Computer-Aided Design*, 1991, pp. 390-393.
- [32] M. del M. Hershenson, S.P. Boyd, and T.H. Lee, “Optimal design of a CMOS op-amp via geometric programming”, *IEEE Transactions on Computer Aided Design*, vol. 20, pp. 1-21, Jan. 2001.
- [33] F. Medeiro *et al.*, “A statistical optimization-based approach for automated sizing of analog cells”, in *Proc. IEEE/ACM International Conference Computer-Aided Design*, 1994, pp. 594–597.
- [34] A. Torralba, J. Chavez, and L. G. Franquelo, “FASY: a fuzzy-logic based tool for analog synthesis”, *IEEE Transactions on Computer Aided Design*, vol.15, pp. 705–715, July 1996.
- [35] M. Krasnicki, R. Phelps, R. Rutenbar, and L. R. Carley, “MAELSTROM: efficient simulation-based synthesis for custom analog cells”, in *Proc. ACM/IEEE Design Automation Conference*, 1999, pp. 945–950.

- [36] W. Kruskamp and D. Leenaerts, "DARWIN: CMOS opamp synthesis by means of a genetic algorithm", in *Proc. ACM/IEEE Design Automation Conference*, 1995, pp. 550–553.
- [37] G. Alpaydin, S. Balkir, and G. Dunder, "An evolutionary approach to automatic synthesis of high-performance analog integrated circuits", *IEEE Transactions on Evolutionary Computation*, vol. 7, pp. 240–252, June 2003.
- [38] F. de Bernardinis, M.I. Jordan, and A. SangiovanniVincentelli, "Support vector machines for analog circuit performance representation", in *Proc. ACM/IEEE Design Automation Conference*, 2003, pp. 964–969.
- [39] C. Makris and C. Toumazou: "ISAID: qualitative reasoning and trade-off analysis in analog IC design automation", in *Proc. IEEE International Symposium on Circuits and Systems*, 1992, pp. 2364-2367.
- [40] M. Barros, G. Neves, J. Guilherme, and N.C. Horta, "A distributed enhanced genetic algorithm kernel applied to a circuit/level optimization e-design environment", in *Proc. Design of Circuits and Integrated Systems*, 2004, pp. 20-24.
- [41] M.Barros, J. Guilherme, and N. Horta, "GA-SVM feasibility model and optimization kernel applied to analog IC design automation", in *Proc. 17th ACM Great Lakes Symposium on VLSI*, pp. 469-472, 2007.
- [42] M.Barros, J. Guilherme, and N. Horta, "Analog circuits optimization based on evolutionary computation techniques", *Integration, The VLSI Journal*, vol. 43, pp. 136-155, January 2010.
- [43] CADENCE IC design environment, Available: <http://www.cadence.com>
- [44] Accellera Organization, "Verilog-AMS language reference manual", June 2009.
- [45] K. Thulasiraman and M. N. S. Swamy, *Graphs: theory and algorithms*, John Wiley & Sons, Inc., 1992.
- [46] S. J. Mason, "Feedback theory — some properties of signal flow graphs", *Proceedings of the IRE*, vol. 41, pp. 1144-1156, September 1953.
- [47] S. J. Mason, "Feedback theory — further properties of signal flow graphs", *Proceedings of the IRE*, vol. 44, pp. 920-926, July 1956.
- [48] C. L. Coates, "Flow graph solutions of linear algebraic equations", in *IRE Transactions on Circuit Theory*, vol. CT-6, pp. 170-187, June 1959.

Application of Symbolic Circuit Analysis for Failure Detection and Optimization of Industrial Integrated Circuits

Ralf Sommer^{1,*}, Dominik Krauß¹, Eric Schäfer¹ and Eckhard Hennig²

¹Ilmenau University of Technology, Germany and ²Institute for Microelectronic and Mechatronic Systems GmbH, Germany

Abstract: The contribution deals with application of symbolic circuit analysis for failure detection and optimization of industrial integrated circuits. It demonstrates how symbolic analysis and approximation allows analyzing industrial analog building blocks systems, which were considered to be symbolically unsolvable before. Besides circuit failure analysis and modeling, a novel methodology that provides a new application-specific compensation for achieving highest performance requirements, is given. The methodology is demonstrated on several industrial examples.

Keywords: Symbolic circuit analysis, circuit failure detection, optimization of industrial integrated circuits, symbolic approximation, frequency behavior, circuit compensation, generalized eigenvalue problem, approximate eigenvalue extraction, poles and zeroes of transfer functions, model order reduction, SBG (simplification before generation), term ranking, eigenvalue sensitivity, Modal Assurance Criterion (MAC), MNA (Modified Nodal Analysis), STA (Sparse Tableau Analysis), small-signal equivalent circuits, BSIM small signal model, industrial CMOS operational amplifier, Analog Insydes, frequency compensation, Nyquist criterion, automated topology modification, direct compensation, compensation network, Analog Insydes.

1. Symbolic Analysis in Industrial IC Design

Analog circuit designers are continuously facing the challenge to squeeze as much performance out of their circuits as the underlying semiconductor technology permits. Pushing a circuit topology to its technological limits requires extensive knowledge of previous designs and deep insight into both the intended functional behavior of the device under construction and the parasitic effects that degrade its performance. At this point computer-aided symbolic analysis can fill a crucial gap in the design process for analog circuits: while traditional numerical circuit simulators fall short in providing qualitative insight into the functional interdependencies between circuit parameters and behavioral characteristics, a symbolic analyzer is capable of deriving such generic knowledge in the form of mathematical expressions automatically from a formal circuit description (netlist) [16, 19]. In this context the most important application area for symbolic analysis is design knowledge acquisition by computer-aided *analytical modeling* of unwanted circuit behavior observed in *numerical simulations*, for example, resonance effects and instability due to parasitic poles [2, 3] or poor power-supply rejection performance (see Section 1.1.).

The preceding statement entails some essential requirements symbolic analysis tools must meet to be useful in industry:

- With the exception of purely educational purposes, the benefit of using a symbolic analyzer in a stand-alone fashion is virtually zero. Symbolic analysis becomes interesting only when applied with the intention to *complement* – and not to replace! – numerical simulation. Thus, in an industrial CAD environment, a

*Address correspondence to Ralf Sommer: Ilmenau University of Technology, Germany; E-mail: ralf.sommer@imms.de

symbolic analyzer must work in tight conjunction with the numerical tools designers use to simulate their circuits. To guarantee consistency of results, the symbolic analyzer must be able to access and process the same data supplied to and generated by the numerical simulators. This requires interfaces for importing circuit schematics, netlists, device model parameter sets, operating-point data, small-signal parameter values, and simulated waveforms directly from the analog circuit simulation environment [7].

- Accurate analytical modeling of simulated effects requires that the symbolic analyzer provides exactly the same semiconductor device models as implemented in the numerical simulator. However, today's device models are very complex, such as in the case of BSIM3 (**Fig. 1**), and their structure is often based entirely on mathematical considerations instead of the underlying geometrical properties of the device. As this makes interpretation of the resulting expressions more difficult, a symbolic analysis tool should also provide a set of calibrated simpler device models, e.g. the SPICE 5-capacitance MOSFET AC equivalent circuit, which may be used in place of the more complicated ones when better interpretability of results is required.
- Due to the extreme complexity of symbolic computation, successful application of symbolic analysis requires a high level of control over the trade-off between modeling accuracy and simplicity. Consequently, a symbolic analyzer must allow the user to add own device models or modify existing ones whenever the problem under investigation demands that particular effects, such as device parasitics, be modeled with special care.

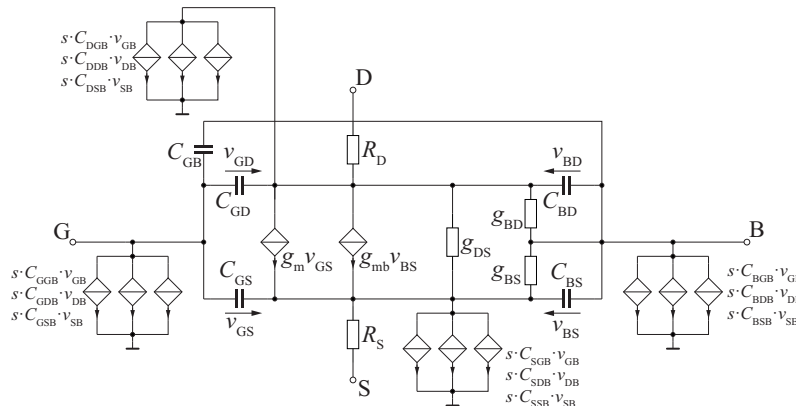


Figure 1: BSIM3 small-signal equivalent circuit.

In the past decade, many symbolic analysis tools have been implemented with emphasis on algorithmic developments [1], but little effort was spent on embedding the systems tightly into industrial circuit simulation environments and offering open device modeling concepts. In addition, experience shows that providing these necessary features is still not sufficient to ensure acceptance of a symbolic analysis tool. Circuit designers are reluctant to use a program whose sole purpose is to read in a circuit description and produce a symbolic transfer function; and the fact that engineers do use general-purpose software with symbolic computation capabilities indicates that the design of symbolic analysis tools has often been approached from the wrong direction:

- Circuit designers employ symbolic computation tools to manipulate, solve, and graph mathematical equations quickly and without human error. In this context,

computing symbolic transfer functions of analog circuits is only one particular task from a wide variety of modeling problems. *A symbolic analyzer should help the designer to perform the circuit modeling tasks more efficiently in a computing environment which he can use to solve other engineering problems as well.* Thus, instead of adding some mathematical and graphical postprocessing functionality to a stand-alone tool, a symbolic analysis program should be implemented as an extension of a widely used general-purpose problem-solving environment, such as Matlab or *Mathematica* [6, 36].

The above point is equally important as the previous requirements because the benefit of automated symbolic analysis becomes apparent to designers only if these techniques can be used in direct conjunction with the vast mathematical, numerical, and graphical processing and programming capabilities offered by the above-mentioned technical computing packages. As a consequence, symbolic analysis tools which do not adhere to this principle have little prospects of success in industry.

1.1. Analysis and redesign of a CMOS operational amplifier

In this section we present a typical example for the use of *Analog Insydes* [4] in industrial circuit design. Consider the CMOS opamp schematic displayed in **Fig 2**. When operated as an inverting amplifier (**Fig. 3**), the circuit exhibits the AC power-supply signal feedthrough (PSF) characteristic shown in **Fig. 4**. At a frequency of 1 kHz, the simulation yields an unexpectedly large PSF value of more than -60 dB while some rough manual calculations done by the designer had predicted this figure to be better than -80 dB. To find the reasons for the discrepancy and improve power-supply rejection, *Analog Insydes* was used to analyze the behavior of the circuit symbolically.

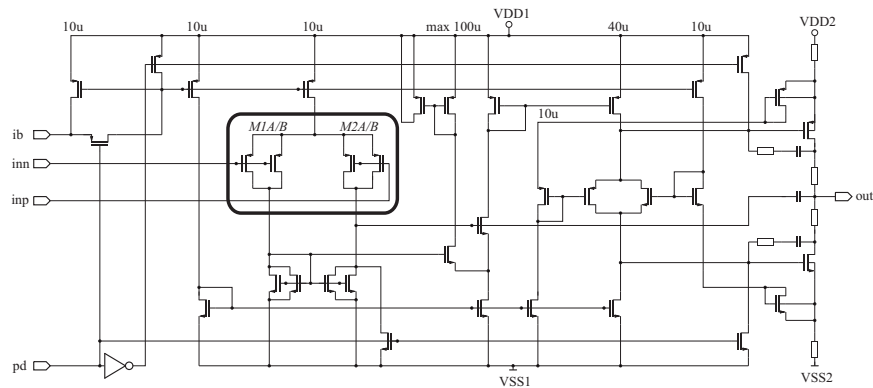


Figure 2: CMOS operational amplifier.

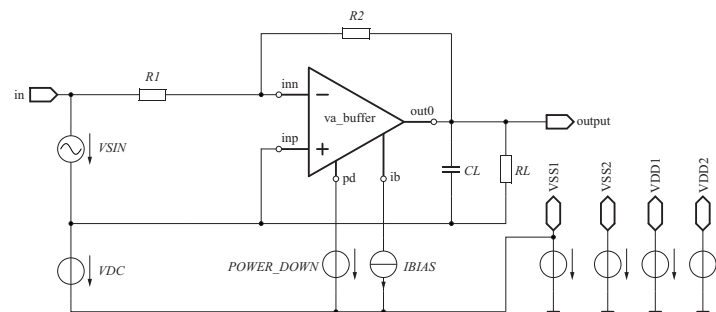


Figure 3: Top-level circuit schematic.

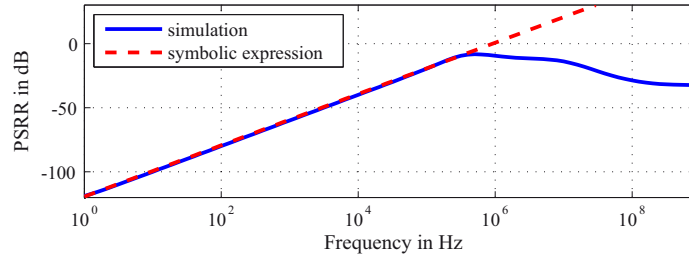


Figure 4: AC power-supply signal feedthrough (simulation: solid line) with Symbolic analysis result (dashed).

After importing all circuit data into *Analog Insydes*, the analysis procedure begins by setting up a system of circuit equations in symbolic form. Using the full BSIM3 AC equivalent circuit for all MOSFET devices (**Fig. 1**), a 43×43 system of modified nodal equations is thus obtained. Solving the equations symbolically for the transfer function from V_{DD} to the output node would result in a huge expression with more than 10^{27} terms. Therefore, *Analog Insydes*' symbolic approximation algorithms are employed to approximate the PSF behavior at low frequencies.

With an error bound of 10% at $f = 1$ kHz, we obtain the following surprisingly simple description of the amplifier's PSF characteristic within only a few seconds of CPU time.

$$V_{\text{outbyVDD}} = \left(-C_{\text{ggbM1A SXI0}} - C_{\text{ggbM1B SXI0}} - C_{\text{gsbM1A SXI0}} \right) \cdot R_2 \cdot s \cdot V_{DD} \quad (1)$$

Taking into account device matching, this expression is further reduced to

$$V_{\text{outbyVDD}} = (-2C_{\text{ggb1}} - C_{\text{gsb1}}) \cdot R_2 \cdot s \cdot V_{DD} \quad (2)$$

where C_{ggb1} and C_{gsb1} denote the gate transcapacitances (**Fig. 1**) of the input devices *M1A* and *M1B* (**Fig. 2**). The approximated expression agrees well with the numerical simulation in the frequency range of interest (**Fig. 4**).

Equation (2) indicates that the high PSF is caused by capacitive coupling through the bulk connection of the input pair, whose area had been made extremely large to reduce mismatch sensitivity and noise. In manual calculations, the designer had neglected the bulk capacitances because he believed them to be insignificant. As a result, an a-priori oversimplification of the problem prevented him from discovering the true reason for the unwanted circuit behavior.

With the knowledge gained from (2), the Power-Supply Rejection (PSR) can be improved by placing the *p*-type input devices together in a separate floating well (**Fig. 5**), thus eliminating the direct capacitive link between bulk and V_{DD} . As shown in **Fig. 6**, this modification improves PSR by more than 40 dB at $f = 1$ kHz. It is interesting to note that the designer had deliberately decided against using a separate well in the initial design to prevent degradation of some performances such as slew rate.

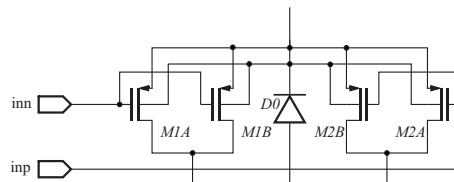


Figure 5: Modified input pair with floating well (well-substrate capacitance modeled through $D0$).

To illustrate the complexity of the problem, after model expansion, the netlist contains 321 primitive components, leading to a 29×29 system of modified nodal equations. Note that the SPICE Level 3 AC model (Fig. 7) has been used instead of the BSIM3 AC model. The differential-mode voltage transfer function has 19 poles and 19 zeros (Fig. 9). In fully expanded symbolic form, it would contain more than 5×10^{19} product terms. To solve the problem and to find out the reason for the peak the pole pair at $s = (-2.1 \pm 8.3j) \times 10^7$ has to be extracted symbolically. The results as well as a solution to the problem will be illustrated in Section 2.4.3.

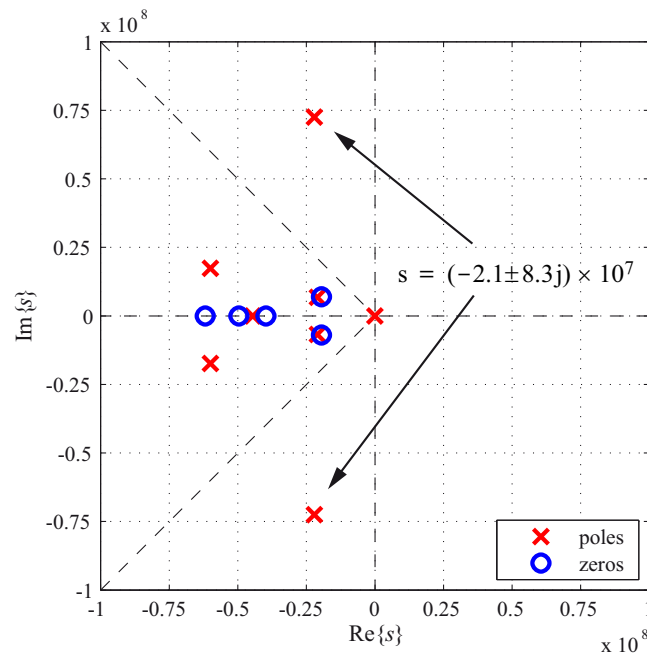


Figure 9: Pole/zero distribution of the original circuit.

1.3. Summary

Most of industrial circuit problems can be rooted back to dynamic behavior. Many problems are related to frequency compensation in conjunction with stability issues. Hence, symbolic extraction of poles and zeroes holds one of the keys for application of symbolic analysis in industrial integrated circuit design.

In the following sections symbolic extraction of formulas for poles and zeroes will be presented. First an extraction methodology based on Simplification Before Generation (SBG) using a magnitude error criterion will be introduced. Due to restrictions of that method to extract compensated pole-zero pairs, which are not visible in the Bode diagram, an eigenvalue sensitivity ranking is developed. This allows SBG to focus on the extraction of a particular eigenvalue independently of its dominance in the frequency response. The extracted formulas will then be used to find appropriate frequency compensations of the circuits under investigation by variation of those circuit parameters that cause those poles to move in such a direction that stability is ensured, and the frequency response does not show peaking effects having a maximum of bandwidth. Finally, an automated topology modification method is developed that introduces additional elements in the circuit in order to optimize frequency behavior. Combined with eigenvalue sensitivity calculations this method allows to find new and unknown variants of compensation networks.

2. Symbolic Pole/Zero Extraction

2.1. Introduction

Symbolic calculation of poles and zeros plays an important role in design-oriented analysis of analog circuits, in particular in the analysis of feedback amplifiers. As the numerator and denominator polynomials of transfer functions are generally irreducible and have degrees much larger than two, it is rarely possible to find mathematically exact expressions for the poles and zeros of an electronic circuit [15, 23]. Consequently, approximations must be introduced to extract at least the technically relevant eigenvalues in analytical form.

2.2. Pole/zero analysis by simplification before generation

In the past several approaches to pole/zero analysis were published that exploit the order reduction effect of SBG methods. The common principle behind these approaches is to find low-order approximations of transfer functions by simplifying systems of circuit equations or determinants with respect to limited frequency intervals. If the frequency response in an interval Δf is characterized by a single real pole or a pair of complex conjugate poles, it is possible to compute local first or second-order approximations of a transfer function from which poles and zeros can be extracted in symbolic form. **Fig. 10** shows a local approximation of a three-pole transfer function $H(s)$ with respect to the frequency interval Δf . Since the characteristics of the frequency response in this interval are determined exclusively by the pole p_2 , the approximation is a transfer function of order 1, which can be solved analytically for p_2 .

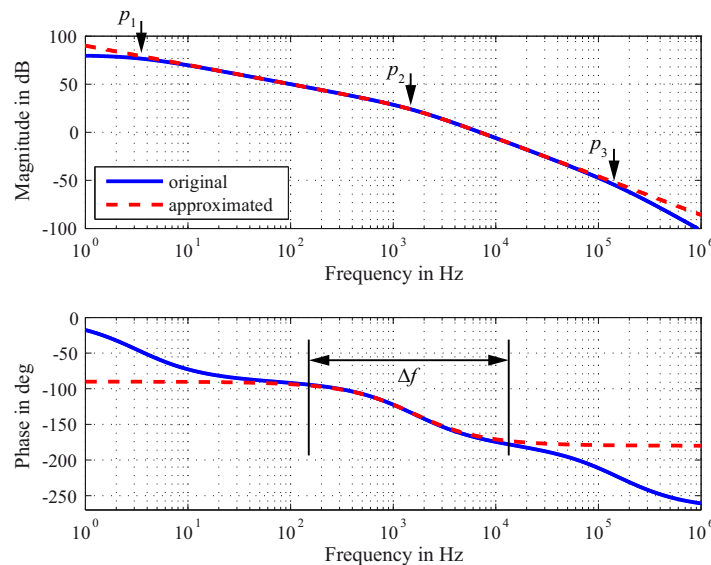


Figure 10: Local approximation (dashed line) of a three-pole transfer.

2.2.1. Matrix-based SBG methods for pole/zero extraction

Matrix-based SBG algorithms for pole/zero analysis were developed by Hsu and Sechen, and by Dröge. Hsu and Sechen use their *sifting approach* to compute local approximations in the neighborhood of corners in the frequency response [25]. If the degrees of the resulting numerator or denominator polynomials are less than three,

then their roots are calculated symbolically. This procedure is applied separately for each frequency interval of interest.

In addition to the SBG component, Dröge's *three-step approximation method* [18, 26], also comprises two Simplification After Generation (SAG) steps [27]. A graphical illustration of the algorithm is shown in **Fig. 11**. In the first step, the algorithm approximates the coefficient matrix of a system of circuit equations

$$\mathbf{T}(s) \cdot \mathbf{x} = \mathbf{b} \quad (3)$$

by discarding terms which have small influence on a selected pole or zero. Expanding the determinant of the approximated matrix then yields a reduced-order characteristic polynomial $P^*(\mathbf{p}, s)$, where \mathbf{p} denotes the vector of symbolic parameters. In the second step, further insignificant terms are removed from the coefficients of P^* . Finally, the roots of the simplified polynomial are calculated symbolically and postprocessed by another SAG step to remove any remaining insignificant contributions.

Approximation of a coefficient matrix \mathbf{T} is driven by a term ranking strategy computed on the basis of root displacements. These displacements are found by evaluating \mathbf{T} with the design-point values \mathbf{p}_0 , expanding the determinant to give a polynomial $P^*(\mathbf{p}_0, s)$ with numerical coefficients, and solving P^* numerically for its roots. This process is very time-consuming as it needs to be repeated for each matrix entry. Therefore, an error estimator based on sensitivities of poles and zeros is provided as a faster though somewhat less reliable alternative to numerical root finding.

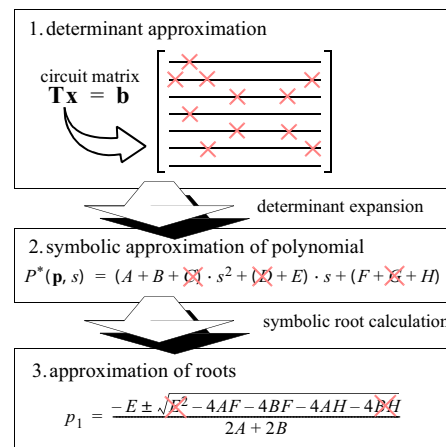


Figure 11: Three-step approximation method.

Both the sifting approach and the three-step approximation method were reported to compare favorably with the pole splitting and zero-value time constant methods in terms of both expression complexity and accuracy. Yet, neither Hsu and Sechen nor Dröge address the potential problem of incorrect eigenvalue pairing that will be discussed in Section 2.3.

2.2.2. Pole/zero extraction by matrix approximation

In [28] Sommer *et al.* discuss a matrix approximation algorithm that uses the large-change influence of matrix entries on the magnitude of a transfer function in one or several design points as term ranking and error control criterion. It is apparent that this algorithm can be applied to pole/zero analysis in a similar fashion as the sifting

approach because both methods are capable of computing local reduced-order approximations.

As illustrated in **Fig. 14** for the case of a common-emitter amplifier (**Fig. 12**), using the simple BJT small signal model shown in **Fig. 13**, a pole or zero can be captured by a suitable choice of two or more design points placed to the left and right of a corner in the magnitude response of a circuit [29]. Here, approximating the system of sparse tableau equations reduces the number of terms in the expanded voltage transfer function of the amplifier from 140 to 9 and the polynomial order of the system from four to two.

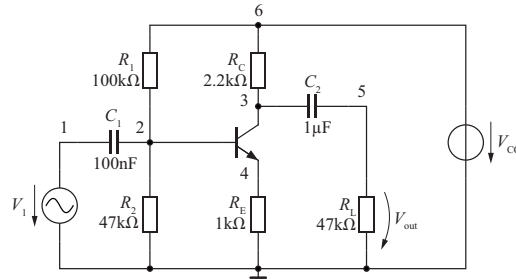


Figure 12: Common-emitter amplifier.

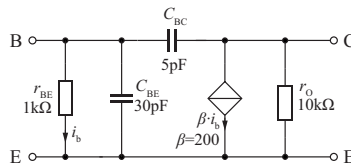


Figure 13: Simple BJT small-signal model.

$$H(\mathbf{p}, s) = \frac{-C_1 C_2 R_1 R_C R_L \beta_{Q1} s^2}{(1 + C_2 R_L s)(R_1 R_2 + R_1 R_E \beta_{Q1} + R_2 R_E \beta_{Q1} + C_1 R_1 R_2 R_E \beta_{Q1} s)} \quad (4)$$

The low-frequency poles and zeros are found immediately from this simplified expression.

$$z_{1,2} = 0 \quad (5)$$

$$p_1 = -\frac{1}{C_2 R_L} \quad (6)$$

$$p_2 = -\frac{R_1 R_2 + R_1 R_E \beta_{Q1} + R_2 R_E \beta_{Q1}}{C_1 R_1 R_2 R_E \beta_{Q1}} \quad (7)$$

Similarly, by placing design points (DP) as shown in **Fig. 14b**, we obtain a high-frequency approximation of the voltage transfer function.

$$H(\mathbf{p}, s) = -\frac{R_C (1 - s C_{BC,Q1} R_E)}{R_E (1 + s C_{BC,Q1} R_C)} \quad (8)$$

Equation (8) yields approximate symbolic expressions for the pole/zero pair that characterizes the frequency response near and beyond the upper end of the passband.

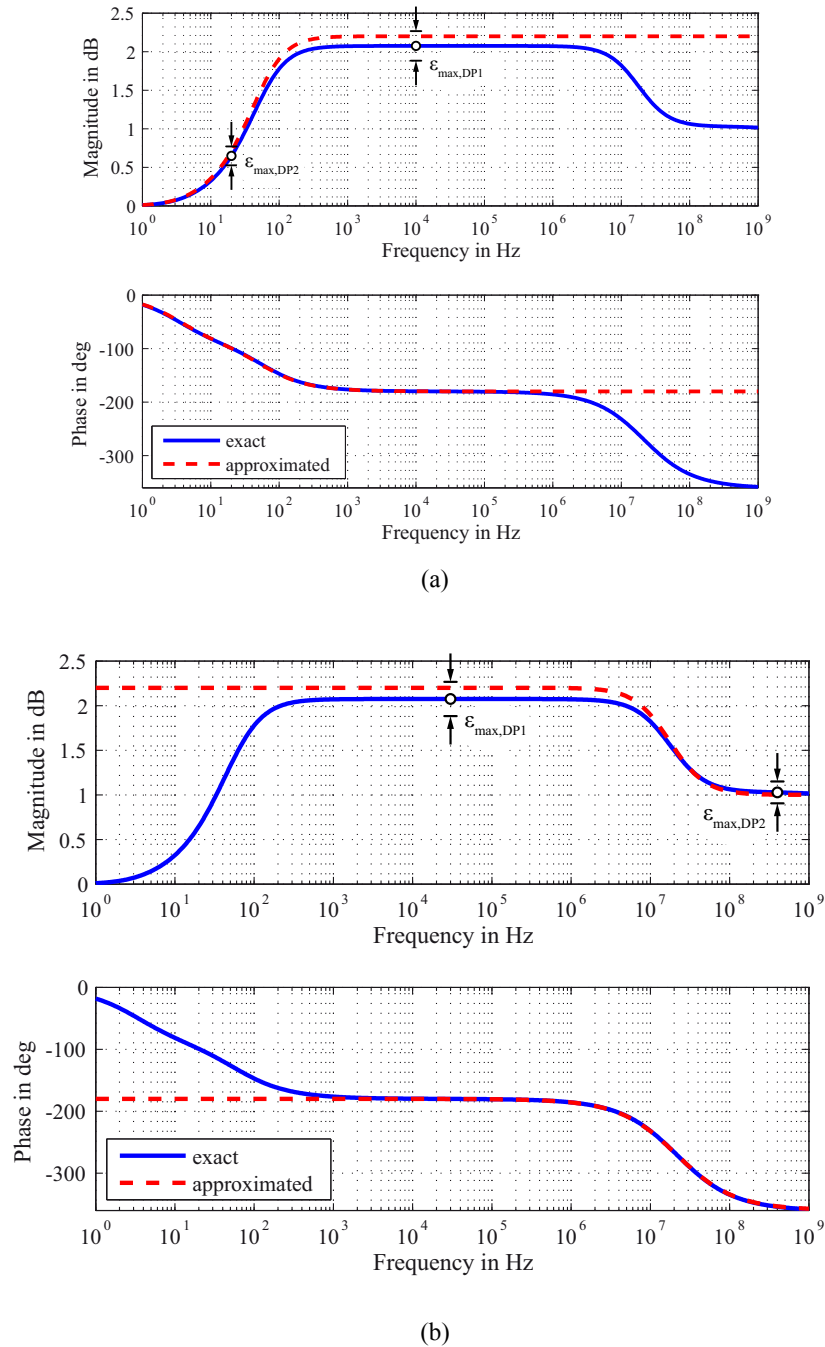


Figure 14: (a) Low-frequency approximation and exact voltage transfer function of the common-emitter amplifier, (b) high-frequency approximation of the common-emitter.

$$z_3 = \frac{1}{C_{BC,Q1}R_E} \quad (9)$$

$$p_3 = -\frac{1}{C_{BC,Q1}R_C} \quad (10)$$

The design-point values of all poles and zeros before and after approximation are listed in **Table 1**.

Table 1: Poles and Zeros of Common-Emitter Amplifier.

Pole/ Zero	Exact Value	Matrix Approximation	Nominal Error	Pole Splitting
z_1, z_2	0.000	0.000	0	0
z_3	1.918×10^8	2.000×10^8	4.3%	2.8%
z_4	-6.948×10^9	N/A	N/A	2.8%
p_1	-2.035×10^1	-2.128×10^1	4.6%	5.1%
p_2	-3.774×10^2	-3.628×10^2	3.9%	5.4%
p_3	-9.509×10^7	-9.091×10^7	4.4%	1.4%
p_4	-6.757×10^9	N/A	N/A	1.4%

Despite being very compact, the approximate expressions are accurate to within 5% of the true numerical values. For comparison, the right-most column lists the nominal errors of pole/zero estimates obtained by applying the pole splitting method to the exact symbolic transfer function. Although the resulting expressions would be much larger, there is little if any gain in accuracy.

Table 1 also reveals a weakness of this matrix approximation approach. Since the degrees of the numerator and denominator polynomials of the exact transfer function are both four, the amplifier has four poles and four zeros. Yet, only three poles and three zeros were found symbolically because the pole/zero pair p_4, z_4 is spaced very closely and thus leaves no visible corners in the frequency response which can be captured. It follows that matrix approximation driven by the magnitude-error criterion allows extracting poles and zeros only if their effect is clearly observable in a Bode diagram.

On the other hand, when this is the case, matrix approximation constitutes an efficient approach to symbolic pole/zero analysis of large circuits. With the development version of Analog Insydes [29], the following formula for the low-frequency open-loop voltage gain of the μ A741 amplifier (**Fig. 15**) can be computed from a 350×350 system of sparse tableau equations in less than 20 CPU seconds. The μ A741 amplifier has a total of 22 zeros and 23 poles, but as the simplified transfer function (11) is only of order 1, we can easily solve its denominator for the dominant pole.

$$H(\mathbf{p}, s) = \frac{g_{m,Q1} g_{m,Q16} g_{m,Q17} g_{m,Q3} g_{m,Q6} (1 + g_{m,Q5} R_1) R_9 r_{O,Q131} r_{O,Q17} r_{O,Q4} r_{BE,Q16} r_{BE,Q17}}{(g_{m,Q1} + g_{m,Q3}) g_{m,Q5} (1 + g_{m,Q6} R_2)} \quad (11)$$

$$\cdot \left[\begin{array}{l} (r_{O,Q131} + r_{O,Q17}) \\ (R_9 r_{O,Q4} + g_{m,Q17} R_8 (r_{O,Q4} + g_{m,Q16} R_9 r_{BE,Q16}) r_{BE,Q17}) \\ + s C_1 g_{m,Q16} g_{m,Q17} R_9 r_{O,Q131} r_{O,Q17} r_{O,Q4} r_{BE,Q16} r_{BE,Q17} \end{array} \right]$$

The nominal value of (12) is $p_1(\mathbf{p}_0) = -19.13$ whereas an exact calculation gives $p_1 = -19.51$. Hence, the nominal approximation error is $\varepsilon_N = 1.9\%$.

$$p_1 = - \frac{(r_{O,Q131} + r_{O,Q17}) (R_9 r_{O,Q4} + g_{m,Q17} R_8 r_{BE,Q17} (r_{O,Q4} + g_{m,Q16} R_9 r_{BE,Q16}))}{s C_1 g_{m,Q16} g_{m,Q17} R_9 r_{O,Q131} r_{O,Q17} r_{O,Q4} r_{BE,Q16} r_{BE,Q17}} \quad (12)$$

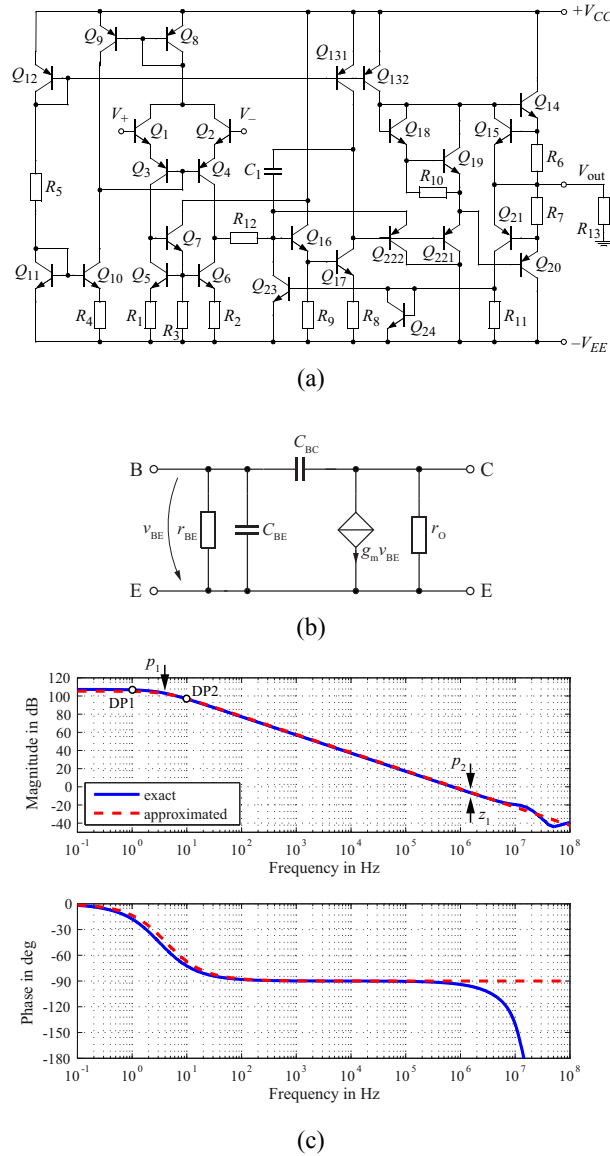


Figure 15: (a) Schematic of the operational amplifier $\mu A741$, (b) Small-signal transistor model for the $\mu A741$ opamp, (c) open-loop gain of the $\mu A741$ amplifier (solid line: SPICE, dashed line: approximated symbolic expression).

By contrast, the opamp's second pole p_2 is compensated by a zero z_1 . We have $p_2 = -6.48 \times 10^6$ and $z_1 = -6.49 \times 10^6$, which corresponds to the frequencies of 1.031 MHz and 1.033 MHz marked in **Fig. 15**. It is apparent from the Bode plot that neither p_2 nor z_1 can be extracted by matrix approximation in the same manner as the dominant pole p_1 because the slope of the magnitude response is nearly constant in the neighborhood of $f = 1$ MHz.

2.2.3. Limitations of the magnitude error criterion

Apart from the above restriction regarding observability, the magnitude-error criterion has several further limitations if used in connection with pole/zero analysis. As even small pole displacements due to matrix approximation may have a large impact on the frequency response of a system, it is often difficult to select sampling points and error bounds such that the error in the approximated pole can be controlled

reliably by monitoring magnitude errors on the imaginary axis or elsewhere in the complex plane. This problem becomes the more obvious the closer a pole is located to the imaginary axis.

The difficulties that arise when using the magnitude-error criterion are illustrated in **Fig. 16** [20]. Assume that we wish to find approximate symbolic expressions for the complex conjugate pair of poles p_1 and p_1^* on the condition that we accept a relatively large deviation of the resonance frequency but allow only a small error in the height of the resonance peak of the transfer function. This implies a large admissible error in the imaginary parts of p_1 and p_1^* . However, the real parts must be computed with higher accuracy to prevent the poles from moving closer to the imaginary axis.

Now assume that a shift of the nominal values of the poles parallel to the imaginary axis to the new positions p_1^* and p_1^* is acceptable. If we specified two sampling points to the left and right of the resonance peak as shown in **Fig. 16**, then the approximation algorithm would stop long before the maximum error in the imaginary parts is reached because a minor shift of the peak would result in excess magnitude error in DP2. On the contrary, an unwanted shift of the poles toward the imaginary axis would remain undetected under the same conditions (**Fig. 16**).

Finally, **Fig. 16** illustrates the case when a global magnitude error causes the approximation algorithm to terminate prematurely although multiplying a transfer function by a constant factor does not change its poles and zeros. This effect can be observed in the case of the $\mu A741$ amplifier.

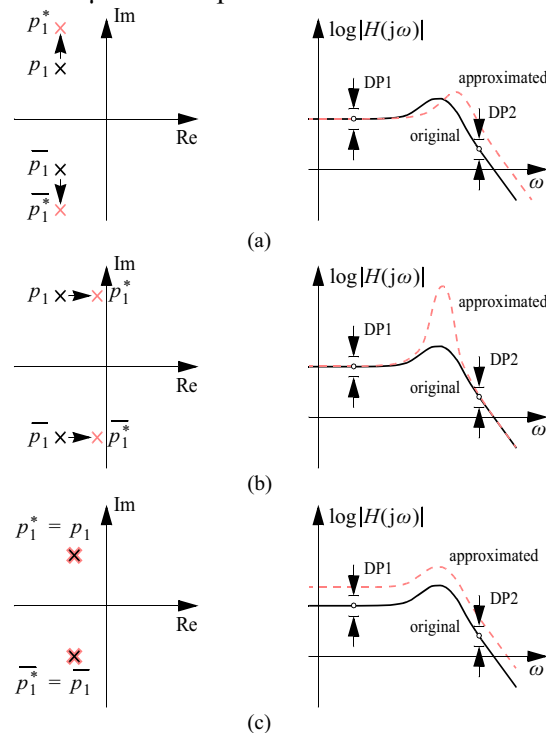


Figure 16: (a) Effect of pole displacement along imaginary axis, (b) effect of pole displacement towards imaginary axis, and (c) effect of global magnitude error.

Here, the symbolic expression for p_1 in (12) is accurate to within $\pm 2\%$ whereas the magnitude errors in the two sampling points are much higher, namely about $1\text{dB} \approx 12\%$.

The problem in **Fig. 16** could be solved by placing additional sampling points on top of the resonance peak and in its neighborhood. Still, the other cases cannot be handled satisfactorily by the same approach because a tight magnitude error envelope only leads to incomplete utilization of the error margins specified for the poles. This is undesirable because the complexity of the resulting symbolic expressions would be higher than necessary for the given error bounds.

2.3. Pole/zero analysis by approximate eigenvalue computation

For improved control of the approximation error and better utilization of error margins it is necessary to use ranking and error tracking criteria which estimate and monitor the shifts of relevant poles and zeros directly. In this section we identify approximate pole/zero analysis with the simplification of a generalized eigenvalue problem and derive a ranking formula for the terms in a symbolic matrix pencil. Then, a new SBG algorithm for pole/zero analysis is formulated on the basis of this ranking approach [12, 38].

2.3.1. The generalized eigenvalue problem

From a theoretical point of view, an ideal starting point for matrix approximation with respect to poles and zeros would be the state-space representation of a linear dynamic system, $(s\mathbf{I} - \mathbf{A})\mathbf{x} = \mathbf{0}$, where \mathbf{A} and \mathbf{x} denote the state matrix and the vector of independent state variables, respectively, and \mathbf{I} is an identity matrix of proper dimension. Indeed, some approaches have been reported that simplify symbolic state equations of electrical circuits using Gershgorin's theory for error estimation [30].

However, except in special cases such as RLC ladders or grids in which the independent state variables can be determined by a simple tree search, state equations cannot be set up directly from a network graph. Instead, they have to be computed from other circuit analysis formulations, *e.g.* nodal analysis, by elimination of dependent variables. Note that the state matrix of a linear circuit constitutes an n -port transfer function and that each matrix entry is given by the DC small-signal transfer function between a port current or voltage and the time derivative of the same or another state variable. Calculating the elements of a state matrix *symbolically* thus entails the computational complexity we seek to avoid in the first place. Therefore, it is generally impossible to obtain the state-space representation of a large circuit in symbolic form.

To compute poles and zeros¹ efficiently by means of matrix-based SBG, it is essential to approximate a system of equations before *any* symbolic computations are performed. Therefore, we rewrite a system of linear circuit equations (3) into the form of a Generalized Eigenvalue Problem (GEP) [31].

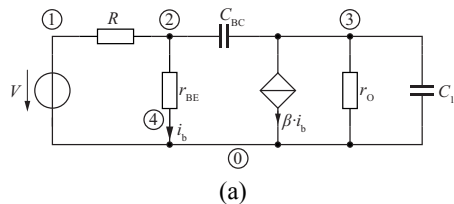
$$\mathbf{T}(s) \cdot \mathbf{x} = (\mathbf{A} - s \cdot \mathbf{B}) \cdot \mathbf{x} = \mathbf{0} \quad (13)$$

Note that the above decomposition requires the coefficient matrix of a system of circuit equations to be a linear function of the Laplace frequency s . Hence, if modified nodal analysis is applied to circuits containing inductors or

¹Since zeros can be determined by solving a subproblem of the GEP (10), all further considerations are, without loss of generality, restricted to the computation of poles.

transimpedances, the matrix fill-in pattern for impedances must be used for these elements to avoid negative powers of s .

To illustrate the representation of any linear circuit as GEP in **Fig. 17** an example circuit and its MNA equations are shown. Subsequently the MNA matrix can be separated into a static and a dynamic part.



$$\begin{pmatrix} \frac{1}{R} & -\frac{1}{R} & 0 & 0 & 1 & 0 \\ -\frac{1}{R} & \frac{1}{R} + \frac{1}{r_{BE}} + s \cdot C_{BC} & -s \cdot C_{BC} & -\frac{1}{r_{BE}} & 0 & 0 \\ 0 & -s \cdot C_{BC} & \frac{1}{r_o} + s \cdot C_1 + s \cdot C_{BC} & 0 & 0 & \beta \\ 0 & -\frac{1}{r_{BE}} & 0 & \frac{1}{r_{BE}} & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \\ I_{V1} \\ i_b \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ V \\ 0 \end{pmatrix}$$

(b)

$$\underbrace{\begin{pmatrix} \frac{1}{R} & -\frac{1}{R} & 0 & 0 & 1 & 0 \\ -\frac{1}{R} & \frac{1}{R} + \frac{1}{r_{BE}} & 0 & -\frac{1}{r_{BE}} & 0 & 0 \\ 0 & 0 & \frac{1}{r_o} & 0 & 0 & \beta \\ 0 & -\frac{1}{r_{BE}} & 0 & \frac{1}{r_{BE}} & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}}_{\mathbf{A}} - s \underbrace{\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -C_{BC} & C_{BC} & 0 & 0 & 0 \\ 0 & C_{BC} & -C_1 - C_{BC} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}}_{\mathbf{B}}$$

(c)

Figure 17: (a) Example circuit to illustrate system equations, (b) corresponding MNA equations of example circuit, and (c) MNA matrix restructured as GEP.

Definition 1

Let $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n \times n}$ and $\lambda \in \mathbb{C}$. The family of matrices $\mathbf{A} - \lambda \mathbf{B}$ is called a *matrix pencil*, denoted by (\mathbf{A}, \mathbf{B}) .

The poles of any transfer function computed from (3) are given by the roots of the characteristic polynomial

$$D(s) = \det(\mathbf{T}(s)) = \det(\mathbf{A} - s\mathbf{B}) \tag{14}$$

To solve the polynomial symbolically for a particular root s_k , its degree is must be less than three²; yet, in general, we have $\deg\{D(s)\} \gg 3$. Now, the roots of $D(s)$ are identical to the solutions s_i of the generalized eigenvalue problem (13). Thus, we seek to compute a reduced-order characteristic polynomial by approximating the matrix pencil (\mathbf{A}, \mathbf{B}) with respect to the eigenvalue of interest λ_k . By discarding all terms in \mathbf{A} and \mathbf{B} which have negligible influence on λ_k , we hope to obtain a first or second-order approximation of expression (16) that allows computing λ_k analytically.

2.3.2. Derivation of eigenvalue sensitivity

To obtain a ranking for the term removal in the matrix pencil a sensitivity information of each parameter with respect to the eigenvalue to be extracted would increase the efficiency of the approximation algorithm very much - comparable for the application of the Shermann-Morrison formula to determine the influence of each matrix entry with respect to magnitude of a selected output variable by only one matrix inversion.

The task is to derive the eigenvalue sensitivities $\partial\lambda/\partial p_i$ of each parameter $p_i \in (\mathbf{A}, \mathbf{B})$ on the solution λ on the corresponding GEP (17), (18) [9].

$$(\mathbf{A} - \lambda\mathbf{B})\mathbf{u} = \mathbf{0} \quad (15)$$

$$\mathbf{v}^H(\mathbf{A} - \lambda\mathbf{B}) = \mathbf{0} \quad (16)$$

where \mathbf{v}^H denotes the Hermitian conjugate of the left eigenvector \mathbf{v} .

A nonzero entry in \mathbf{A} or \mathbf{B} consists of a single term or a sum of terms, for example $G_1 + R_L^{-1} - gm_{Q1}$, $C_2 + C_L$ or $1 + \beta_{CM2}$. Each single term or part of a sum shall be referred to as a (*device*) *parameter*. Assume that \mathbf{A} and \mathbf{B} contain a total of m parameters $p_1, \dots, p_m \in \mathbb{R}$. Hence, the eigenvalues and eigenvectors are functions of a device parameter vector $\mathbf{p} := (p_1, \dots, p_m)^T \in \mathbb{R}^m$, i.e. $\lambda = \lambda(\mathbf{p})$, $\mathbf{u} = \mathbf{u}(\mathbf{p})$, $\mathbf{v} = \mathbf{v}(\mathbf{p})$.

Differentiating (17) with respect to an arbitrary parameter p_i yields

$$\frac{\partial(\mathbf{A} - \lambda\mathbf{B})\mathbf{u}}{\partial p_i} = \left(\frac{\partial\mathbf{A}}{\partial p_i} - \frac{\partial\mathbf{B}}{\partial p_i} \right) \mathbf{u} - \frac{\partial\lambda}{\partial p_i} \mathbf{B}\mathbf{u} + (\mathbf{A} - \lambda\mathbf{B}) \frac{\partial\mathbf{u}}{\partial p_i} = \mathbf{0}. \quad (17)$$

Obviously, the first derivatives of \mathbf{A} and \mathbf{B} exist with respect to all p_i and are given by $\partial\mathbf{A}/\partial p_i = \mathbf{A}_i$ and $\partial\mathbf{B}/\partial p_i = \mathbf{B}_i$ and are easily to be determined since they contain elements only in linear form so that the derivative will have only 1 and -1 on the position of the parameter or term as defined above.

However, although λ and \mathbf{u} are generally differentiable, there may be some sets of parameter values $\mathbf{p}_0 \in \mathbb{R}^m$ for which $\partial\lambda/\partial p_i$ and $\partial\mathbf{u}/\partial p_i$ do not exist [9]. Therefore, we assume that \mathbf{p}_0 is chosen appropriately as to ensure existence of all derivatives.

Premultiplying (19) by the corresponding left eigenvector \mathbf{v}^H gives

$$\mathbf{v}^H \left(\frac{\partial\mathbf{A}}{\partial p_i} - \lambda \frac{\partial\mathbf{B}}{\partial p_i} \right) \mathbf{u} - \frac{\partial\lambda}{\partial p_i} \mathbf{v}^H \mathbf{B}\mathbf{u} + \mathbf{v}^H (\mathbf{A} - \lambda\mathbf{B}) \frac{\partial\mathbf{u}}{\partial p_i} = \mathbf{0}. \quad (18)$$

²Although the roots of polynomials with a degree up to four can be found analytically the resulting expressions are usually very complex and uninterpretable if the degree is greater than two.

Because $\mathbf{v}^H(\mathbf{A} - \lambda\mathbf{B}) = 0$, we obtain the following expression for the sensitivity $\partial\lambda/\partial p_i$ of the eigenvalue λ with respect to a change in the value of the parameter p_i .

$$\frac{\partial\lambda}{\partial p_i} = (\mathbf{v}^H\mathbf{B}\mathbf{u})^{-1} \mathbf{v}^H \left(\frac{\partial\mathbf{A}}{\partial p_i} - \lambda \frac{\partial\mathbf{B}}{\partial p_i} \right) \mathbf{u} \tag{19}$$

Note that $\mathbf{v}^H\mathbf{B}\mathbf{u} \neq 0$ for $\lambda \neq \infty$ [33].

To determine a ranking of all matrix entries (21) is applied to each term in A and B. **Table 2** shows an example for such a ranking.

Table 2: Example for a Term Ranking Using Eigenvalue Sensitivity Formula (16).

Rank	Matrix	Row	Col	p_i	$\ \Delta\lambda\ /\ \lambda\ $
1	B	1	1	$-C_1$	1.55×10^{-30}
2	A	6	6	$1/R_1$	2.71×10^{-30}
3	A	6	6	$1/R_C$	1.23×10^{-28}
...					
27	B	3	3	$-C_2$	0.0205858
28	B	3	5	C_2	0.0218158
...					

Once the ranking is determined term removal starts with the terms according the ranking list. Since the ranking only reflects the influence on the eigenvalue shift of the individual term, the deletion of more than one term in the order of the ranking will have to be tracked by recalculation of the eigenvalues. When approximating a system of circuit equations with respect to a single pole or zero, only one eigenpair (λ, \mathbf{u}) of the corresponding GEP needs to be tracked numerically. Approximation errors can thus be calculated more efficiently with an iterative GEP solver than with the QZ algorithm, which always computes the entire spectrum of the GEP [37]. Moreover, if the error induced by eliminating a parameter from (\mathbf{A}, \mathbf{B}) is small, an iterative method should converge to the corresponding eigenpair $(\lambda^*, \mathbf{u}^*)$ of the perturbed GEP very rapidly if the design-point value $(\lambda_0, \mathbf{u}_0)$ of the original eigenpair is supplied as initial guess. Conversely, if the method fails to converge, then the eigenvalue shift is too large. In both cases a small number of iterations should be enough to either correct the eigenpair or recognize that a perturbation is inadmissible. A suitable method for this purpose is the *Jacobi Orthogonal Correction Method (JOCM)* [5, 11].

2.3.3. The eigenvalue pairing problem

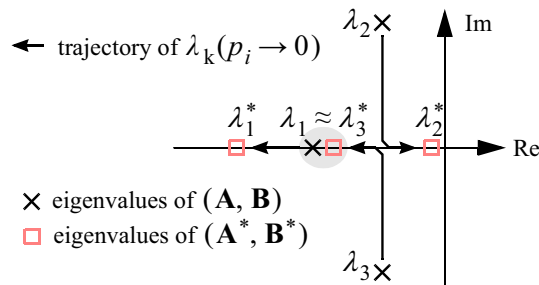


Figure 18: The eigenvalue pairing problem.

A central task of the SBG algorithm is to determine whether eliminating a particular parameter p_i is admissible for a given error bound ε_{\max} . Assume that $(\lambda_1, \mathbf{u}_1)$ is an eigenpair of (17) where λ_1 denotes the eigenvalue we wish to extract, and that $\lambda_k^*, \mathbf{u}_k^*$ is an eigenpair of the perturbed GEP

$$(\mathbf{A}^* - \lambda \mathbf{B}^*) \mathbf{u}^* = \mathbf{0} \quad (20)$$

obtained by setting a parameter $p_i \in (\mathbf{A}, \mathbf{B})$ to zero. If (22) is a valid approximation of the original GEP with respect to λ_1 , and if $(\lambda_k^*, \mathbf{u}_k^*)$ and $(\lambda_1, \mathbf{u}_1)$ are corresponding eigenpairs, we have

$$\lambda^* \approx \lambda \Leftrightarrow \text{error}(\lambda_1, \lambda_k^*) < \varepsilon_{\max} \quad (21)$$

However, condition (23) alone is not sufficient to ensure that λ_k^* is indeed identical to the element λ_1^* of the spectrum of $(\mathbf{A}^*, \mathbf{B}^*)$ which corresponds to λ_1 . In fact, λ_k^* may just be some other eigenvalue of the perturbed pencil that has incidentally moved to a position close to λ_1 while λ_1^* has been shifted far away from its original position λ_1 . This situation is illustrated in **Fig. 18**, where λ_3^* could be falsely associated with λ_1 because the trajectory of $\lambda_3(p_i)$ ends near λ_1 as p_i approaches zero. It follows that a simple error measure based only on distances between eigenvalues does not constitute a reliable criterion for determining whether a parameter may be discarded.

False pairing of eigenvalues can be avoided by applying the *Modal Assurance Criterion* (MAC), which constitutes a measure for the correlation of two eigenvectors [10]. The MAC between two eigenvectors \mathbf{u} and \mathbf{u}^* is defined as

$$\text{MAC}(\mathbf{u}, \mathbf{u}^*) := \frac{|\mathbf{u}^H \mathbf{u}^*|^2}{(\mathbf{u}^H \mathbf{u})(\mathbf{u}^{*H} \mathbf{u}^*)}. \quad (22)$$

The value of the MAC ranges from 0 to 1. A value of 1 means that the vectors are multiples of each other; a value of 0 means that they are orthogonal. For corresponding eigenpairs $(\lambda_k^*, \mathbf{u}_k^*)$ and $(\lambda_1, \mathbf{u}_1)$, \mathbf{u}_1 and \mathbf{u}_k^* should be closely correlated; hence, $\text{MAC}(\mathbf{u}_1, \mathbf{u}_k^*)$ must be very close to 1. Otherwise λ_k^* cannot be considered a valid approximation of λ_1 even if (23) holds.

To ensure that an eigenvalue λ_k^* of the approximated GEP is a valid approximation of the original eigenvalue λ_1 , the error criterion for accepting a device parameter elimination must be augmented as follows:

$$\text{error}(\lambda_1, \lambda_k^*) < \varepsilon_{\max} \wedge \text{MAC}(\mathbf{u}, \mathbf{u}_k^*) > m_{\min}, \quad (23)$$

where m_{\min} denotes a real number close to 1. Following in Section 2.3.4., it will be shown that the MAC can be integrated easily into the error control process when used in combination with an iterative GEP solver. The complete algorithm is outlined in **Fig. 19**.

2.3.4. Pole/zero extraction algorithm with fast error tracking

The input parameter λ denotes a rough numerical estimate of the eigenvalue of interest and is used only to select the correct element λ_0 from the spectrum of $(\mathbf{A}_0, \mathbf{B}_0)$. The error bound ε_{\max} does not necessarily have to be a single real number. In the case of a complex eigenvalue $\lambda \in \mathbb{C} \setminus \mathbb{R}$, ε_{\max} may also represent a pair of two separate error bounds $(\varepsilon_r, \varepsilon_i)$, one for the real part of λ and one for the imaginary part. Alternatively, it could be supplied as an application-specific function that uses ε_{\max} as a control parameter.

The value of the control parameter r_{\max} must be selected very carefully depending on the numerical condition of \mathbf{A}_0 and $\lambda_0 \mathbf{B}_0$. If the maximum residual norm is too large, then the error tracking procedure will signal convergence even if (ϑ, \mathbf{y}) is not remotely a solution of $(\mathbf{A}^*, \mathbf{B}^*)$. On the contrary, the procedure may fail to converge to a valid eigenpair as a result of numerical inaccuracies if r_{\max} is chosen too small. Typical values of r_{\max} used to obtain the results discussed in were of the order 10^{-8} to 10^{-10} for the common-emitter amplifier and 10^{-8} to 10^{-16} for the larger examples.

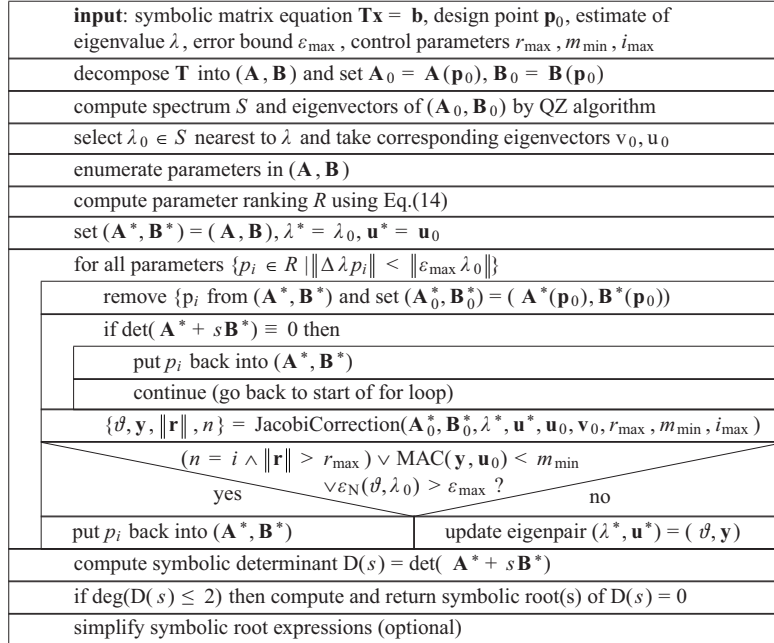


Figure 19: Flowchart of pole/zero extraction algorithm with iterative error tracking.

The MAC acceptance threshold m_{\min} is less critical because the value of the MAC usually remains either very close to 1 or tends to decrease rapidly and significantly if the Jacobi corrections diverge. Therefore, a default value of $m_{\min} = 0.98$ generally constitutes an appropriate choice. Similarly, the maximum number of iterations i_{\max} is an uncritical parameter. In most cases, convergence is obtained within two or less iterations if a perturbation causes an acceptable error. Thus $i_{\max} = 3 \dots 6$ commonly suffices to distinguish non-convergent sequences of iterates from convergent ones. Larger values only increase the computation time spent on the non-convergent cases and rarely lead to better results.

After a parameter ranking has been computed by means of, the approximation loop by default selects those parameters p_i as candidates for removal whose estimated influence $\|\Delta\lambda\|$ is less than the fraction ε_{\max} of the magnitude of the nominal eigenvalue $\|\lambda_0\|$, *i.e.* $\|\Delta\lambda(p_i)\| < \|\varepsilon_{\max} \lambda_0\|$. The p_i from the list of candidates are then removed from the matrix pencil $(\mathbf{A}^*, \mathbf{B}^*)$ one by one. Before calling the error tracking function, we first check if the pencil has become singular and undo the most recent perturbation if this is the case.

Although the eigenvalue of interest may not be affected by singularity, the situation where $\det(\mathbf{A}^* + s\mathbf{B}^*) \equiv 0$ must be avoided for obvious reasons. Most singularity effects have structural reasons, that is when discarding a parameter leads to a zero row or column. These cases can be identified quite easily by keeping track of the

number of nonzero entries in each row and column of $\mathbf{A}^* + s\mathbf{B}^*$. On the contrary, mathematically reliable detection of non-structural singularity cannot be accomplished without computing $\det(\mathbf{A}^* + s\mathbf{B}^*) \equiv 0$ symbolically. As this approach is clearly impractical, we perform a rough numerical quasi-singularity check by LU decomposition of $\mathbf{A}_0^* + s\mathbf{B}_0^*$ for some $s \in \mathbb{C}$ that is not an eigenvalue of the pencil.

Following the singularity check, the error tracking procedure is called to compute the true shift of the eigenpair $(\lambda^*, \mathbf{u}^*)$. If the algorithm converges within i_{\max} iterations, and if both the MAC and the error specifications are respectively satisfied for \mathbf{u}^* and λ^* , the perturbation is accepted. Otherwise, the parameter p_i is put back into the matrix pencil. Then, the procedure is repeated for the next parameter.

As soon as all parameters have been processed, the determinant of the approximated matrix $\mathbf{T}^* = \mathbf{A}^* + s\mathbf{B}^*$ is computed symbolically. Provided that all other eigenvalues of (\mathbf{A}, \mathbf{B}) are located sufficiently far apart from λ , the polynomial degree of $\det(\mathbf{T}^*)$ can be expected to be one if λ is real or two if λ is complex. If this is true, the polynomial can be solved symbolically for its roots. In an optional postprocessing step, the resulting root expressions may be simplified further as shown in **Fig. 11**.

2.4. Examples

2.4.1. Analysis and compensation of a CMOS differential amplifier

As a concluding example, consider the CMOS differential amplifier shown in **Fig. 20** [14], Problem 12.10. Some symbolic analysis results for poles and zeroes are presented in Chapter 11 (“Symbolic pole/zero analysis”). Suppose now that the dimensions of M_9 are not chosen appropriately to cancel the zero in the right half plane introduced by the Miller capacitance C_C ; for example, assume that $W/L(M_9) = 50\mu/8\mu$. Thus, under the given loading conditions, the small-signal transfer function from v_+ to v_{out} has three dominant eigenvalues (two poles and one zero) whose numerical values are listed in **Table 3**. In the following we extract these eigenvalues symbolically and use the results to compensate the amplifier.

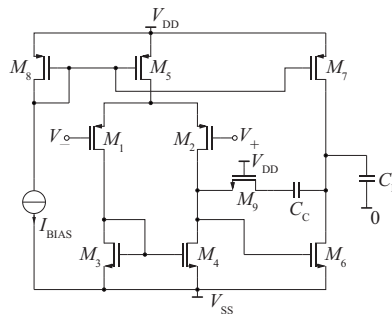


Figure 20: CMOS differential amplifier.

Table 3: Dominant Poles and Zeros of the CMOS Operational Amplifier.

P/Z	Original	Approximated	Error Bound	Nominal Error
z_1	2.508×10^7	2.548×10^7	5%	1.6%
p_1	-5.260×10^3	-5.510×10^3	5%	4.8%
p_2	-1.042×10^7	-1.063×10^7	5%	2.0%
		-1.130×10^7	10%	7.8%

Using the MOS transistor model in **Fig. 21** leads to a 14×14 system of modified nodal equations with six finite generalized eigenvalues (poles). With an error bound of $\varepsilon_{\max} = 5\%$, the approximation algorithm computes the following symbolic expressions for p_1 and p_2 from the MNA matrix.

$$p_1 = -\frac{(g_{DS,M2} + g_{DS,M4})(g_{DS,M6} + g_{DS,M7})}{C_L(g_{DS,M2} + g_{DS,M4}) + C_C g_{m,M6}} \quad (24)$$

$$p_2 = -\frac{(C_C C_{GS,M6} + C_C C_L + C_{GS,M6} C_L) G_{DS,M9}}{2C_C C_{GS,M6} C_L} \quad (25)$$

$$-\frac{\sqrt{(C_C C_{GS,M6} + C_C C_L + C_{GS,M6} C_L)^2 g_{DS,M9}^2 - 4C_C^2 C_{GS,M6} C_L g_{DS,M9} g_{m,M6}}}{2C_C C_{GS,M6} C_L}$$

A further simplification of p_2 can be achieved with a less restrictive error specification. With $\varepsilon_{\max} = 10\%$, we obtain

$$p_2 = -\frac{g_{m,M6}}{C_L}. \quad (26)$$

The right half-plane zero is calculated as

$$z_1 = \frac{G_{DS,M9} g_{m,M6}}{C_C (G_{DS,M9} - g_{m,M6})}. \quad (27)$$

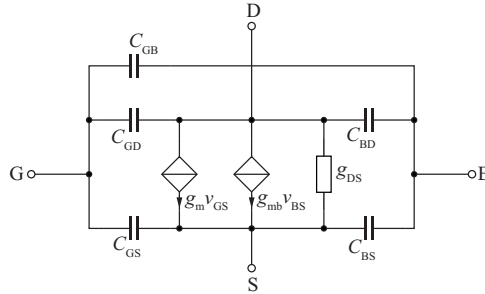


Figure 21: MOS transistor AC model.

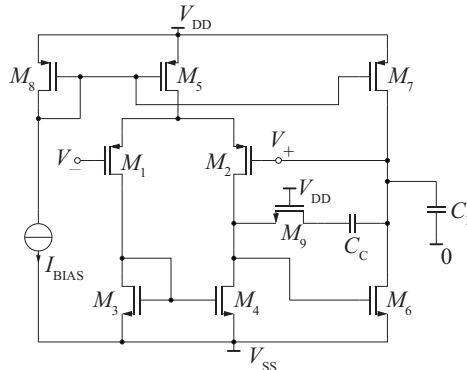


Figure 22: CMOS differential amplifier in unity-gain feedback configuration.

Equation (30) shows that the zero can be cancelled, or at least moved to sufficiently high frequencies, by adjusting $W/L(M_9)$ such that $G_{DS,M9} \approx g_{m,M6}$. Now assume that $W/L(M_9)$ has been dimensioned accordingly and that the amplifier is operated in unity-gain feedback configuration with an increased load capacitance of $C_L = 50\text{pF}$

as shown in **Fig. 22**. In this configuration, the circuit has a dominant complex conjugate pair of poles $p_{1/2}$ which causes a resonance peak in the frequency response near $f = 1$ MHz (**Fig. 24**):

$$p_{1/2} = -0.998 \times 10^6 \pm j5.911 \times 10^6 \quad (28)$$

Although a single approximation run suffices to reduce the determinant of the MNA matrix to a second-order polynomial with only six terms, we apply the algorithm a second time after matrix compression to obtain a more interpretable expression for p_1 . The nominal value of the approximated pole pair is $p_{1/2,0} = -0.810 \times 10^6 \pm j6.262 \times 10^6$.

$$p_1 = \overline{p_2} = -\frac{g_{m,M2}g_{m,M6}}{2C_L G_{DS,M9}} + \frac{\sqrt{g_{m,M2}g_{m,M6}(-4C_L G_{DS,M9}^2 + C_C g_{m,M2}g_{m,M6})}}{2\sqrt{C_C C_L G_{DS,M9}}} \quad (29)$$

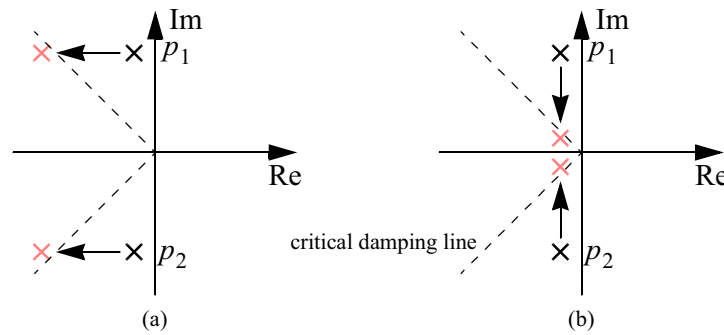


Figure 23: Frequency compensation by moving the poles beyond the line of critical damping.

To compensate the unity-gain follower, the poles must be moved into the 90° sector delimited by the line of critical damping shown in **Fig. 23**. This can be accomplished either by shifting the poles away from the imaginary axis (**Fig. 23(a)**), or by shifting the poles along the imaginary axis towards lower frequencies, thereby reducing bandwidth (**Fig. 23(b)**).

For the given load C_L , (29) suggests trying alternative (a) by decreasing $G_{DS,M9}$ and increasing $g_{m,M6}$. In order to attain critical damping, we seek to multiply the real parts of the poles by a factor of about 10, for instance, by doubling $g_{m,M6}$ and dividing $G_{DS,M9}$ by 5. These modifications can be achieved by doubling the widths of M_6 and M_7 , and by increasing the length of M_9 to $50 \mu\text{m}$. A comparison of the frequency responses before and after compensation as computed by SPICE is shown in **Fig. 24**. Note that compensation according to **Fig. 24(a)** increases the bandwidth of the circuit from 1.4 MHz to 3.5 MHz at the expense of increasing the quiescent power consumption from $693 \mu\text{W}$ to $941 \mu\text{W}$. Moreover, the chip area of the modified circuit is slightly larger.

Equation (32) also indicates that the unity-gain follower can be compensated according to **Fig. 24(b)**. Since the pole pair is complex, it follows that the argument of the right-most square root function in the numerator must be negative:

$$-4C_L G_{DS,M9}^2 + C_C g_{m,M2}g_{m,M6} < 0 \quad (30)$$

Hence, the imaginary part of p_1 can be reduced by increasing the value of the capacitance C_C , which constitutes the classical approach to frequency compensation.

A numerical pole/zero analysis yields that a capacitance $C_C = 75$ pF is required to obtain critical damping. The SPICE simulations displayed in **Fig. 25** confirm this result. However, apart from the reduction of bandwidth from 1.4 MHz to 270 kHz, this alternative is practically infeasible because the value of the compensation capacitance is unreasonably large for integrated-circuit fabrication. On the other hand, a (theoretical) advantage of this approach is a higher load independence. For alternative (b), C_L may assume any value from zero to 50 pF after compensation, whereas alternative (a) does not permit to vary the load by more than 10 pF in either direction without causing significant ripple in the frequency response. Still, this example shows that computer-aided symbolic circuit analysis can help to find alternative solutions to circuit design problems that may have advantages over conventional approaches in special applications.

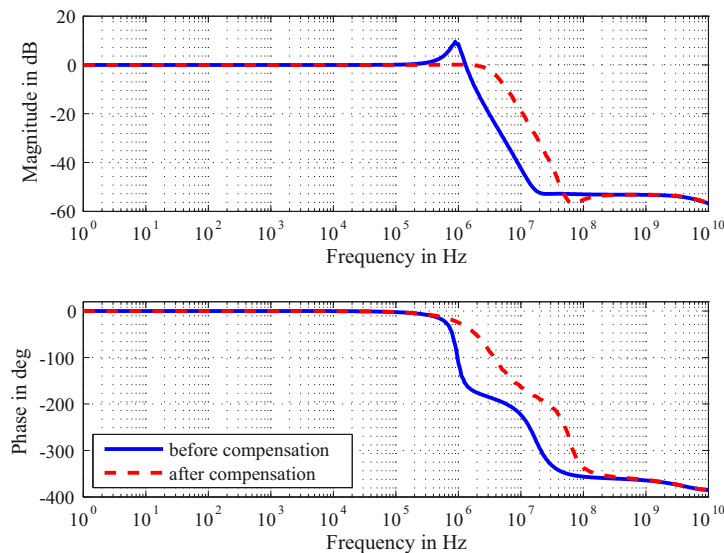


Figure 24: Frequency response before (solid line) and after compensation (dashed line).

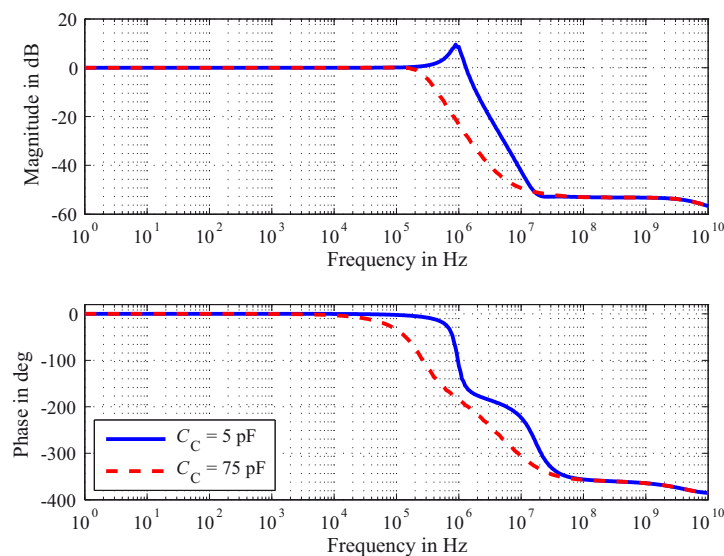


Figure 25: Frequency responses for $C_C = 5$ pF (solid line) and $C_C = 75$ pF (dashed line).

2.4.2. Instability and frequency compensation problem of CMOS folded-cascode operational amplifier

In the first step of the analysis procedure, the netlist is read into the symbolic analysis tool including all necessary model, operating-point, and small-signal parameters. For better interpretability of the resulting symbolic expressions, the MOS devices were modeled using the SPICE Level 3 AC model (**Fig. 2**) instead of the BSIM3 AC model.

After model expansion, the netlist contains 321 primitive components, leading to a 29×29 system of modified nodal equations. The differential-mode voltage transfer function has 19 poles and 19 zeros (**Fig. 9**). In fully expanded symbolic form, it would contain more than 5×10^{19} product terms. To extract the pole pair at $s = (-2.1 \pm 8.3j) \times 10^7$ symbolically, the algorithm outlined in the next sections has been applied yielding the following symbolic formula:

$$-\frac{(C_{C0} + C_L)g_{m\$MN6}}{2C_{C0}C_L} \pm \frac{\sqrt{C_{gs\$MP15}g_{m\$MN6} \left(C_{gs\$MP15} (C_{C0} + C_L)^2 g_{m\$MN6} - 4C_{C0}^2 C_L g_{m\$MP15} \right)}}{2C_{C0}C_L C_{gs\$MP15}}. \quad (31)$$

The extracted formula reveals that for a given load C_L , compensation capacitance C_{C0} , and fixed operating points, the value of the gate-source capacitance of the device MP15 should be increased in order to decrease the value of the imaginary parts of the pole pair; this can be achieved by adding a shunt capacitor between the gate and source terminals of MP15.

Fig. 27 shows a root locus plot of the amplifier calculated from the original (unsimplified) system of equations as $C_{GS,MP15}$ is swept from 1 pF to 10 pF. The plot shows that the above conclusion drawn from the approximated symbolic pole expression is valid.

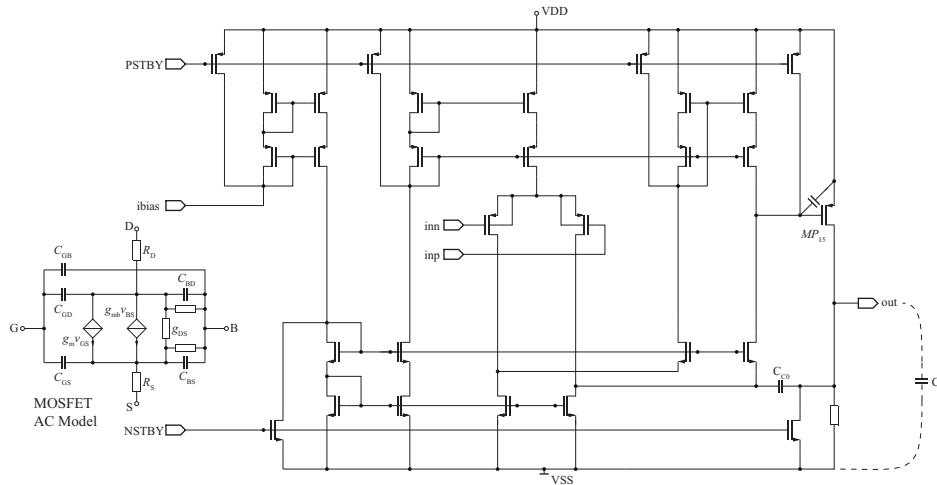


Figure 26: CMOS folded-cascode operational amplifier with compensation capacitor derived from symbolic analysis.

3. Frequency Compensation by Automated Topology Modification

3.1. Introduction and classical approach

Frequency compensation of feedback amplifier circuits is usually achieved by adjusting the open-loop frequency response such that the phase-margin criterion is fulfilled, *e.g.* by shifting the dominant pole to lower frequencies.

The open-loop design method is a universal approach to compensating general-purpose operational amplifiers (op-amps). As a result, amplifier bandwidth may be reduced more than necessary because some extra safety margin must be provided. Furthermore, for the design of stable high-performance circuits with respect to gain and bandwidth, the conventional means of feedback theory are limited: parasitic effects of several groups of transistors form internal feedback loops, even in open-loop configuration. As a result, the open-loop approach may no longer be applicable. Thus, it is often insufficient to check stability with the phase-margin criterion only. On the other hand, direct analysis and design of closed-loop systems is more difficult because their transfer functions are harder to derive and often involve complex poles and zeros.

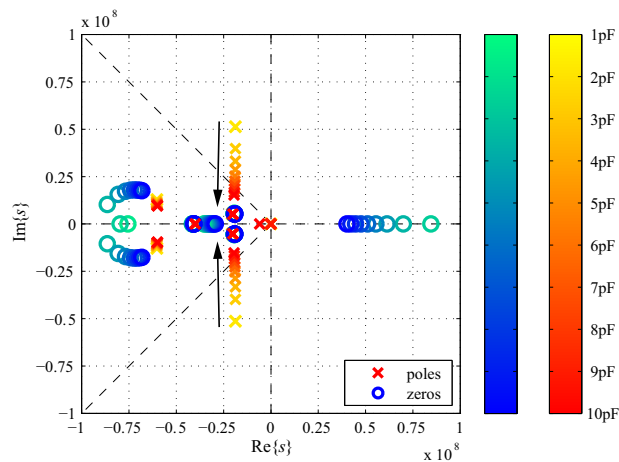


Figure 27: Root locus analysis of the voltage transfer function as $C_{GS,MP15}$ is swept from 1 pF to 10 pF.

In feedback configuration, open-loop real valued poles often turn into complex pole pairs causing ringing in the circuit's transient response. In addition, internal local loops in the circuit topology often lead to clusters of several pole-pairs such that the frequency response is no longer determined by a simple dominant pole configuration.

As illustrated in the previous sections the aim of direct frequency compensation is to change the locations of poles and zeros of the circuit's transfer function in feedback configuration such that maximum bandwidth is obtained. The strategy outlined before was based on generating approximated formulas for poles and zeros by symbolic circuit analysis. These formulas express the poles and zeros as functions of dominant parameters. It can be concluded from manual interpretation of these formulas which parameters have to be modified in order to move the eigenvalue of interest into the desired direction. Most of the terms contained in the extracted formulas are transistor small-signal parameters. Hence, eigenvalues can be moved by changing the operating point. However, this is often inconvenient because all AC parameters are coupled. As a consequence, it is normally not possible to change just one single small-signal parameter at a time. Moreover, changing an operating point of one transistor will influence the entire DC configuration of the circuit. To avoid this, modifications were restricted to adding external capacitances in parallel to small-signal device capacitances. Although this methodology was limited to the amplification of already existing circuit parameters, it introduced the concept of topology modification. This motivated a generalization to a systematic exploration of topological circuit modifications that do not change the DC conditions.

3.2. Approach to systematic circuit topology modification

The idea of amplifying already existing small-signal parameters by adding elements to the respected nodes can be generalized in the following manner: New branches are inserted to the circuit by connecting each pair of nodes with a capacitor or a series of a resistor and a capacitor. This prevents the operating points to change due to the inserted elements. For a circuit with n independent nodes

$$N = \frac{n \cdot (n-1)}{2} \quad (32)$$

additional capacitances are inserted. The idea is illustrated in **Fig. 28**.

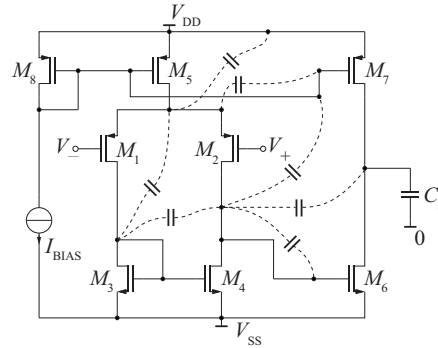


Figure 28: Illustrating example for automatically inserted branches (not all combinations shown).

In the second step, root locus plots are generated for each inserted capacitance. Then those capacitances are automatically selected that allow the poles and zeros to be shifted such that the frequency response is improved with respect to peaking, bandwidth, etc.

$$\mathbf{B} = \begin{matrix} & i & j \\ i & \begin{pmatrix} \vdots & \vdots \\ \cdots & -C_k & \cdots & C_k & \cdots \\ \vdots & \vdots \\ j & \cdots & C_k & \cdots & -C_k & \cdots \\ \vdots & \vdots \end{pmatrix} & \end{matrix} \quad \frac{\partial \mathbf{B}}{\partial p_k} = \begin{matrix} & i & j \\ i & \begin{pmatrix} \vdots & \vdots \\ \cdots & -1 & \cdots & 1 & \cdots \\ \vdots & \vdots \\ j & \cdots & 1 & \cdots & -1 & \cdots \\ \vdots & \vdots \end{pmatrix} = (\mathbf{e}_i - \mathbf{e}_j) \otimes (\mathbf{e}_j - \mathbf{e}_i)$$

Figure 29: Example for derivate of MNA pattern for a capacitance C_k in the \mathbf{B} matrix of matrix pencil (\mathbf{A}, \mathbf{B}) .

To obtain the eigenvalue sensitivity of a capacitance, (21) in Section 2.3.2 is used according to **Fig. 29**. On the left side in **Fig. 29** the pattern of a capacitance in (13) is shown in modified nodal analysis formulation. In contrast to the definition of a parameter in Section 2.3.2 a parameter p_k refers to a complete capacitance and not only to a single element of an MNA pattern. The derivative of $\mathbf{B}(\mathbf{p})$ with respect to one parameter $p_k = C_k$ is given by the right part of **Fig. 29**, where \otimes denotes the Kronecker product. All other entries are 0 since they do not depend on p_k . It follows that for the calculation of eigenvalue sensitivities with respect to the inserted capacitances at the nominal point $C(p = 0)$ only the connectivity information for each capacitance is necessary. No further assumptions about absolute or relative numerical relationships among the capacitance values are required. In this context, “positive

influence” is defined as shifting a complex eigenvalue in such a way that ringing/peaking is decreased (**Figs. 30 and 31**) [21].

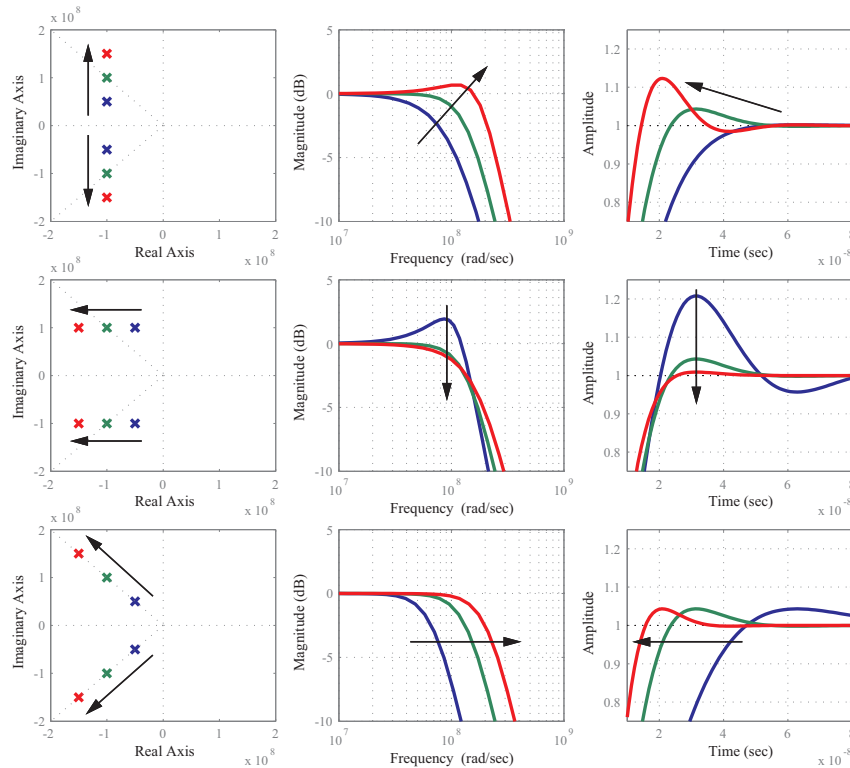


Figure 30: Association between Pole-Zero Map, Transient Response and Bode Plot

This means that the absolute value of the real part has to be larger than the imaginary part (gray area depicted in **Fig. 31**) in the left half plane (stable region). It should be noted that there are often various possibilities for selecting and modifying capacitances such that a desired effect in the frequency response is obtained. This is caused by sets of capacitances that have similar influence on the eigenvalues of interest.

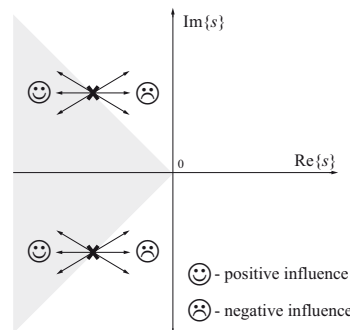


Figure 31: Illustrating example for definition of influence on relevant poles.

In the next step the introduced capacitances have to be sized. This is done by the numerical sizing and optimization tool WiCkeD [39]. **Fig. 33** shows the complete flow of the automated frequency compensation based on topology modification.

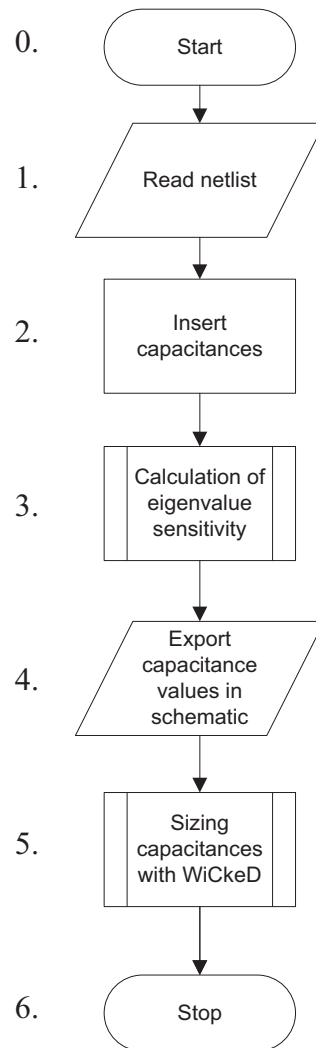


Figure 32: Flowchart of the automated compensation approach.

3.3. Example of an industrial application

The presented approach has been successfully applied to a transimpedance amplifier (TIA, Fig. 33) used in an industrial Optoelectronic Integrated Circuit (OEIC) design.

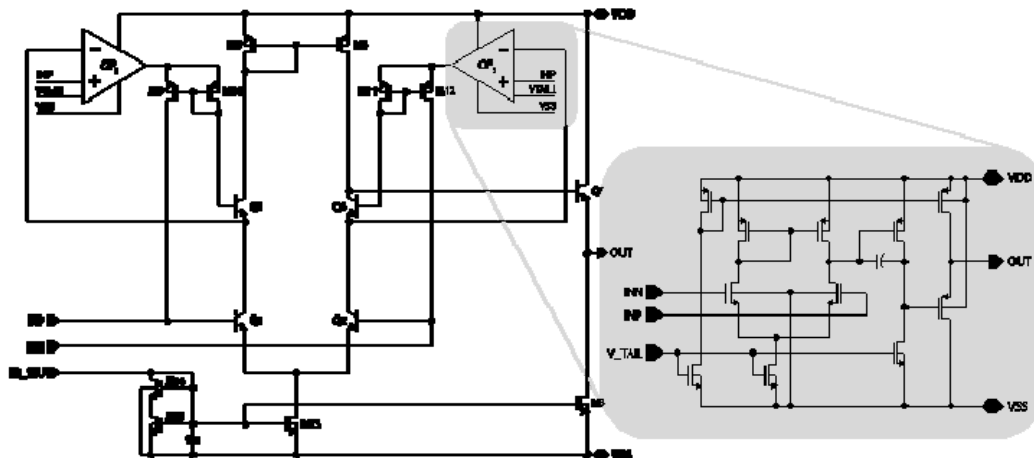


Figure 33: Schematic of high-performance transimpedance amplifier without compensation.

These classes of amplifiers have to provide high bandwidth and slew rate, low offset, and low peaking in the closed-loop frequency response. To determine the characteristics of the TIA, the testbench in **Fig. 34** was used.

The challenge is to get a very fast circuit by introducing complex pole pairs. We allow for some overshoot in the transient response to improve the slope of the output signal.

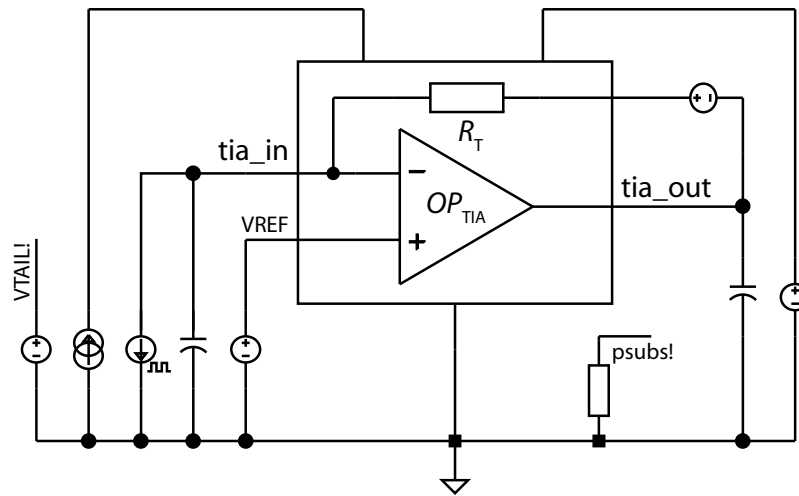


Figure 34: Top-Level of the TIA.

To obtain good matching of the input transistors, BJTs are used in the first stage. Base current cancellation is applied to improve the offset. Due to the low supply voltage of 5 V, controlling op-amps (OP_1 , OP_2 in **Fig. 33**) are used to ensure that Q_1 - Q_4 do not leave the active region. In **Fig. 35** the transient response of the TIA (**Fig. 33**) is shown. As a ringing can be observed the circuit has to be compensated next.

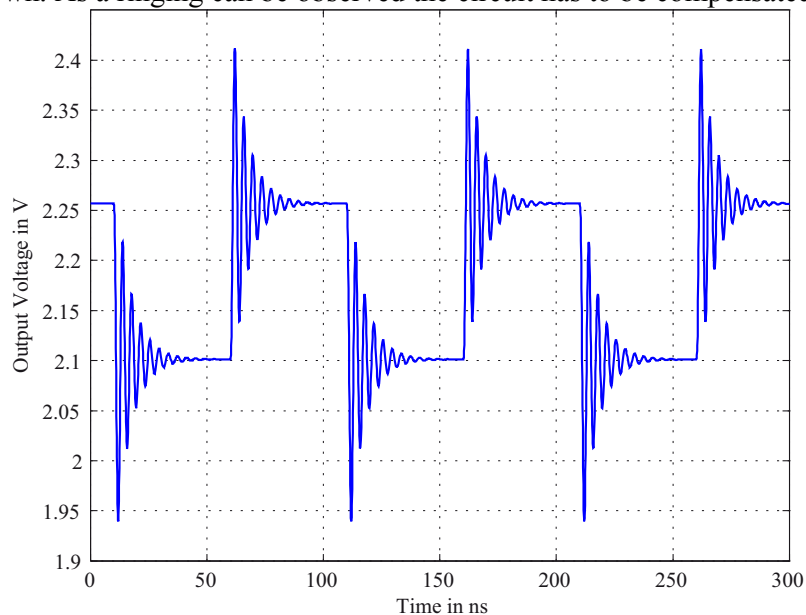


Figure 35: Transient response of uncompensated TIA.

After application of the step 0–4 in **Fig. 32** the 9 capacitances displayed in **Fig. 36** have been selected out of originally more than 600 between each pair of nodes. These are now transferred to Cadence Virtuoso Schematic Editor.

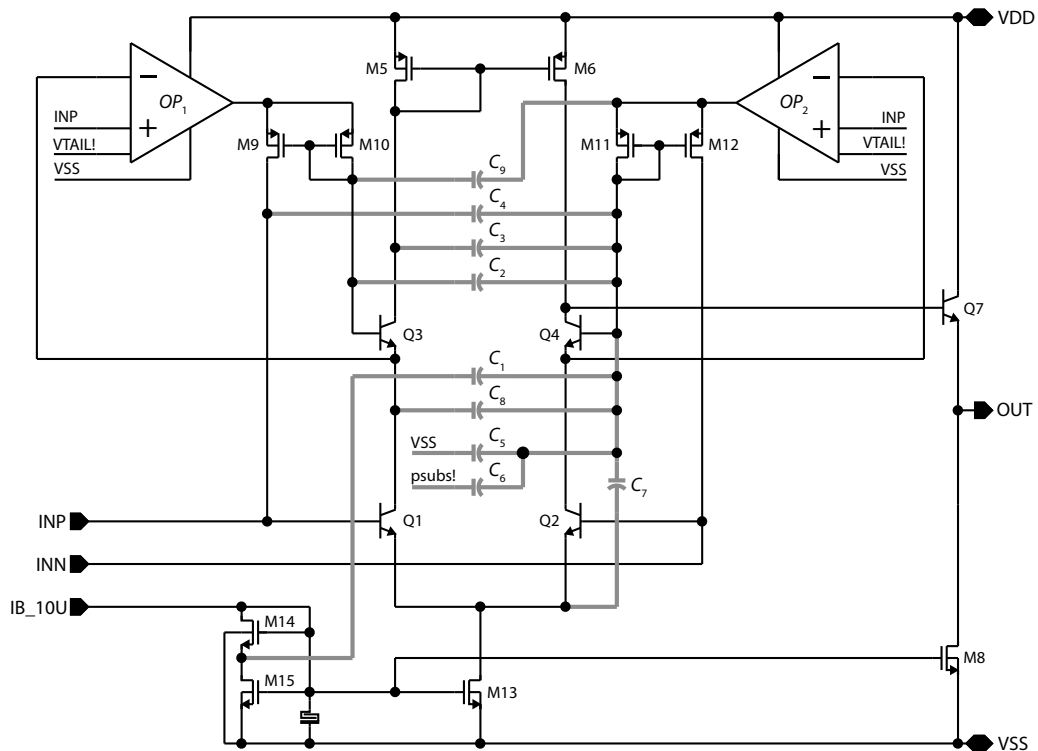


Figure 36: TIA with new capacitances inserted after eigenvalue sensitivity calculation.

The next steps include nominal sizing and yield optimization that are performed by WiCkED.

In summary, we obtain the results shown in **Table 4** for the capacitances and the performances in **Table 5**.

Table 4: Sized and Optimized Capacitances.

Capacitance	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
Value	393 fF	0	87 fF	385 fF	0	424 fF	0	0

Table 5: Performances of TIA without Compensation in Comparison with the Results After Application of the Compensation Algorithm.

Performance	3 dB-Bandwidth	Peaking	Slewrate Rising	Slewrate Falling
without compensation	471 MHz	16.6 dB	213 V/ μ s	1600 V/ μ s
WiCkED optimized	525 MHz	0.98 dB	202 V/ μ s	537 V/ μ s

Fig. 37 illustrates the final transient response, **Fig. 38** comparison of poles and zeroes before and after compensation of the TIA, and **Fig. 39** displays the frequency responses before and after compensation. It should be noted that the 3dB-bandwidth of the compensated TIA is even larger than for the uncompensated circuit. Moreover, as the circuit is yield optimized the final yield is 99.95%.

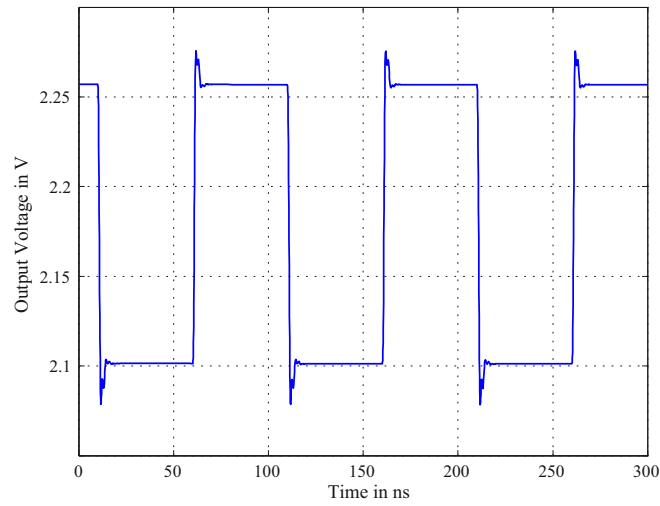


Figure 37: Transient response of compensated TIA.

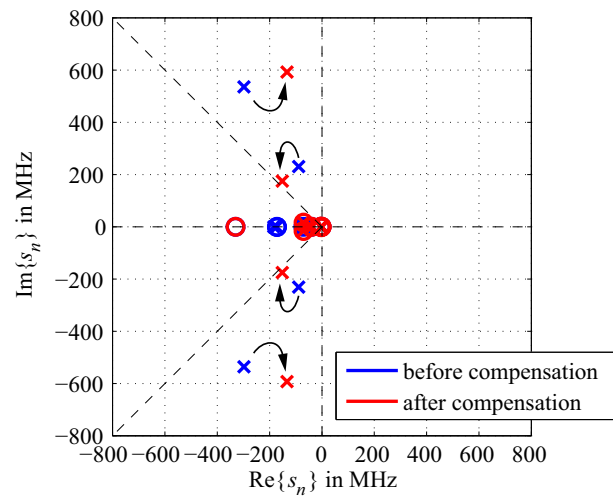


Figure 38: Comparison of poles and zeroes before and after compensation.

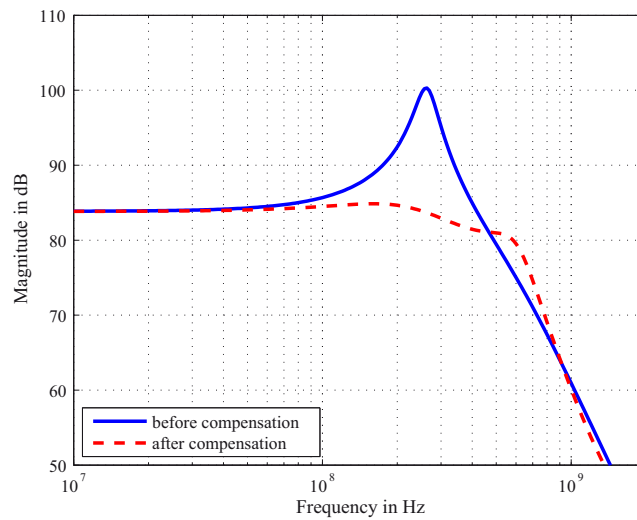


Figure 39: Comparison of frequency response before and after compensation.

4. Conclusion

It has been shown that a lot of industrial circuit design problems can be rooted back to dynamic behavior. Symbolic analysis, especially the symbolic extraction of poles and zeroes allows the designer to identify those circuit elements that are responsible for the related frequency behavior. The interpretation of the obtained formulas may then help to modify the elements in such a way, that the circuit gets stable, a larger bandwidth or less ringing. One of the key issues here is to investigate and adapt the circuit in its actual feedback and load configuration. This direct compensation method often yields much more performance of the circuit compared to classical compensation approaches from feedback theory that try to increase open-loop phase- and gain-margin. Finally, direct circuit compensation has been extended with an automated topology modification that introduces new elements to the circuit. As these modifications should not change the operating point new branches consisting of capacitances are inserted, which allow for changing the location of poles and zeroes in such a way that a maximum of frequency performance is obtained. Using eigenvalue sensitivity calculations to find the appropriate topology modifications combined with numerical optimization techniques this method allows to find new and unknown variants of compensation networks for high-performance amplifier IC designs.

References

- [1] F. V. Fernández, A. Rodríguez-Vázquez, J. L. Huertas, and G. Gielen, *Symbolic analysis techniques – applications to analog design automation*, IEEE Press, New York, 1998
- [2] E. Hennig, R. Sommer, “A reliable iterative error tracking method for approximate symbolic pole/zero analysis”, in *Proc. European Conference on Circuit Theory and Design*, vol. 1, pp. 193–196, Helsinki, Aug. 2001
- [3] E. Hennig, *Symbolic approximation and modeling techniques for analysis and design of analog circuits*, Shaker Verlag, Aachen, 2000
- [4] Analog Insydes: www.analog-insydes.de
- [5] R. Sommer, E. Hennig, M. Thole, T. Halfmann, T. Wichmann, “Analog insydes 2 – new features and applications in circuit design”, in *Proc. International Workshop on Symbolic Methods and Applications to Circuit Design*, Lisbon, Oct. 2000
- [6] S. Wolfram, *The mathematica book*, 3rd ed., Wolfram Media/Cambridge University Press, 1996
- [7] U. Feldmann, R. Schultz, “TITAN: a universal circuit simulator with event control for latency exploitation”, in *Proc. Solid-State Device Research Conference (ESSCIRC'88)*, Manchester, 1988, pp. 183–185
- [8] E. Hennig, R. Sommer, M. Wiese, “Approximate symbolic pole/zero extraction using equation-based simplification driven by eigenvalue shift prediction”, in *Proc. International Symposium on Circuits and Systems*, Monterey, 1998, pp. 25 – 28
- [9] F. Kuhnert, “Über die sensibilität von eigenwerten und eigenvektoren”, *ZAMM*, vol. 71, no. 7/8, 1991, pp. 233–239
- [10] M. I. Friswell and I. E. Mottershead, *Finite element model updating in Structural mechanics*, Kluwer Academic Publishers, Dordrecht, 1995
- [11] G. L. Sleijpen, A. G. L. Booten, D. R. Fokkema, H. A. V. d. Vorst, “Jacobi-davidson type methods for generalized eigenproblems and polynomial eigen problems: part I”, *BIT Numerical Mathematics*, vol. 36:3, pp. 595–633, 1996
- [12] E. Hennig, *Symbolic approximation and modeling techniques for analysis and design of analog circuits*, Shaker Verlag, Aachen, 2000
- [13] E. Hennig, R. Sommer, “Frequency compensation of closed-loop feedback amplifier systems”, in *Proc. International Symposium on Circuits and Systems*, 2000, vol.3, pp. 121–124
- [14] P. R. Gray and R. G. Meyer, *Analysis and design of analog integrated circuits*, 2nd ed., Wiley, New York, 1984
- [15] F. V. Fernández, A. Rodríguez-Vázquez, and J. L. Huertas, “A tool for symbolic analysis of analog integrated circuits including pole/zero extraction”, in *Proc. 10th European Conference on Circuit Theory and Design*, Copenhagen, Denmark, 1991, pp. 752–761

- [16] M. Amadori, R. Guerrieri, and E. Malavasi, "Symbolic analysis of simplified transfer functions", *Analog Integrated Circuits and Signal Processing*, vol. 3, no. 1, pp. 9–29, 1993
- [17] H. Floberg, and S. Mattison, "Computer-aided symbolic circuit analysis CASCA", *Alta Frekvenza*, vol. 5, no. 6, pp. 24–28, Dec. 1993
- [18] G. Dröge, *Symbolische verfahren zur entwurfsunterstützung analoger schaltungen*, Shaker, Aachen, 1997
- [19] G. Nebel, *Symbolische analyse analoger BiCMOS-schaltungen*, Fortschritt-Berichte VDI, Reihe 9, Nr. 213, VDI-Verlag, Düsseldorf, 1995
- [20] A. S. Sedra and K. C. Smith, *Microelectronic circuits*, 2nd ed., Holt, Rinehart and Winston, New York, 1987
- [21] W. C. Elmore, "The transient response of damped linear networks with particular regard to wideband amplifiers", *Journal of Applied Physics*, vol. 19, no. 1, pp. 55–63, Jan. 1948
- [22] S. B. Haley and P. J. Hurst, "Pole and zero estimation in linear circuits", *IEEE Transactions on Circuits and Systems*, vol. CAS-36, no. 6, pp. 838–845, June 1989
- [23] J. D. Rodríguez-García, O. Guerra, F. V. Fernández, and A. Rodríguez-Vázquez, "Symbolic pole/zero extraction through dedicated SBG/SDG techniques", in *Proc. 5th International Workshop on Symbolic Methods and Applications to Circuit Design*, pp. 43–48, Oct. 1998
- [24] O. Guerra, J. D. Rodríguez-García, F. V. Fernández, A. Rodríguez-Vázquez, "A symbolic pole/zero extraction methodology based on analysis of circuit Time-constants", *Analog Integrated Circuits and Signal Processing*, 31, pp. 101–118, Kluwer Academic Publishers, The Netherlands, 2002
- [25] J. J. Hsu and C. Sechen, "Accurate extraction of simplified symbolic pole/zero expressions for large analog IC's", in *Proc. IEEE International Symposium on Circuits and Systems*, Seattle, USA, May 1995, pp. 2083–2087
- [26] G. Dröge, T. Czysz, and E.-H. Horneber, "Symbolic pole and zero estimation for circuit design", in *Proc. 3rd IEEE International Conference on Electronics, Circuits, and Systems*, Rodos, Greece, 1996, pp. 93–96
- [27] G. Gielen and W. Sansen, *Symbolic analysis for automated design of analog integrated circuits*, Kluwer Academic Publishers, Boston, 1991
- [28] R. Sommer, E. Hennig, G. Dröge, and E.-H. Horneber, "Equation-based symbolic approximation by matrix reduction with quantitative error prediction", *Alta Frekvenza*, vol. 5, no. 6, pp. 29–37, Dec. 1993
- [29] E. Hennig and T. Halfmann, *Analog insydes tutorial*, ITWM, Kaiserslautern, Germany, 1998
- [30] F. Constantinescu and M. Nitescu, "A new approach to symbolic pole computation", in *Proc. European Conference on Circuit Theory and Design*, Istanbul, Turkey, 1995, pp. 655–658
- [31] J. Vlach and K. Singhal, *Computer methods for circuit analysis and design*, 2nd ed., Van Nostrand Reinhold, New York, 1994
- [32] C. B. Moler and G. W. Stewart, "An algorithm for generalized matrix eigenvalue problems", *SIAM Journal on Numerical Analysis*, vol. 10, pp. 241–256, 1973
- [33] G. W. Stewart, *Matrix perturbation theory*, Academic Press, Boston, 1990
- [34] M. I. Friswell and I. E. Mottershead, *Finite element model updating in Structural dynamics*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1995
- [35] G. L. Sleijpen, A. G. L. Booten, D. R. Fokkema, and H. A. V. d. Vorst, "Jacobi-davidson type methods for generalized eigenproblems and polynomial eigenproblems: Part I", University Utrecht, Preprint no. 923, Nov. 1995
- [36] S. Wolfram, *The Mathematica book*, 3rd ed., Wolfram Media/Cambridge University Press, 1996
- [37] Y. Drexelmeier, T. Wichmann, E. Hennig, and R. Sommer, "An improved generalized eigenvalue solver for circuit analysis", in *Proc. 5th International Workshop on Symbolic Methods and Applications to Circuit Design*, Kaiserslautern, Germany, 1998, pp. 36–38
- [38] E. Hennig, J. M. Tweer, and R. Sommer, "Enhanced symbolic matrix approximation techniques", in *Proc. 5th International Workshop on Symbolic Methods and Applications to Circuit Design*, Kaiserslautern, Germany, 1998, pp. 199–206
- [39] R. Schwenker, F. Schenkel, H. Grab, K. Antreich, "The generalized boundary curve - a common method for automatic nominal design and design centering of analog circuits. In *Proc. Design and Test in Europe*, Paris, France, 2000, pp. 42–47

