uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Tamer Ahmed

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

M.A.Sc. (Electrical Engineering)

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

An Adjoint-based Approach to Computing Time-domain Sensitivity of Multi-port Systems Described by Reduced-order Models

TITRE DE LA THÈSE / TITLE OF THESIS

M. Yagoub

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

E.Gad

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

D. McNamara

Q.J. Zhang

Gary W. Slater

Le Doyen de la Faculté des études supérieures et postdoctorales / Dean of the Faculty of Graduate and Postdoctoral Studies

An Adjoint-Based Approach to Computing Time-Domain Sensitivity of Multi-port Systems Described by Reduced-Order Models

by

Tamer Ahmed

Thesis submitted to the

Faculty of Graduate and Postdoctoral Studies

In partial fulfillment of the requirements

For the M.A.Sc. degree in

Electrical and Computer Engineering

School of Information Technology and Engineering

Faculty of Engineering

University of Ottawa

© Tamer Ahmed, Ottawa, Canada, 2006

Library and Archives Canada

Published Heritage Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et Archives Canada

Direction du Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 978-0-494-25738-8
Our file Notre référence
ISBN: 978-0-494-25738-8

NOTICE:
The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

AVIS:
L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

Abstract

The present thrust in the electronics industry towards integrating multiple functions on a single chip while operating at very high frequencies has highlighted the need for efficient Electronic Design Automation (EDA) tools to shorten the design cycle and capture market windows. One of the fundamental tasks performed by modern EDA tools is the sensitivity analysis, which is concerned with computing the derivatives of various circuit performance metrics w.r.t. the circuit designable parameters.

However, the increasing complexity of modern circuit design has made simulation and sensitivity analysis a computationally cumbersome task. Furthermore, the presence of non-linearity in virtually all designs in the form of buffers, driving circuitry, and analog blocks necessitates that these tasks be carried out in the time domain.

This thesis describes a new algorithm to compute the time-domain sensitivity using the concept of Model-Order Reduction (MOR). The proposed algorithm enables an interconnected set of large linear systems, that have been abstracted by reduced-order models for the purpose of simulation, to have their sensitivity evaluated in the time domain. Therefore, the key advantage of the proposed work stems from the fact that it brings the computational efficiency of MOR in areas of simulation to bear on the problem of sensitivity analysis.

The large systems are assumed to be interconnected within a larger network using lumped linear or nonlinear components, and the desired sensitivity is to be evaluated w.r.t. their internal designable parameters. The proposed algorithm can thus be applied in computing the sensitivity of circuits constructed from microwave devices, such as multi-conductor transmission lines, which when discretized produce large linear systems.

The proposed technique also employs the concept of adjoint networks in the sensitivity computation. In addition it can also be used to obtain the sensitivity information in the frequency domain.

Acknowledgements

I would like to express my sincere gratitude to my thesis supervisor, prof. Emad Gad whom with his help this work would not have come to this completion. Thank you...

I would like to extend my sincere thanks to prof. Mustapha Yagoub for his support during the work of this thesis.

I would also like to thank professor R. Achar for his patience with his student and for the clear and the quality of the material he presented to us.

Prof P. Gunupudi has presented us with interesting topics and also sparking ideas for new research problems. He is capable for sure of making class material interesting, Thank you...

I have also to mention my classmates with whom I spent very good time understanding lot of new material presented in class, quality time chatting about life and life matters! I can not enumerate all of them here, however, everyone has added more dimension to the aspect of knowledge through the diverse culture we all enjoyed. Thank you all for being part of this work!

Although coming at the end but first in mind is my family and friend that either provided emotional and social support for me and pushed me forward during hard times.

Pages iv-v missing.

Contents

List of Figures

Notation

$\boldsymbol{\mathcal{A}}$	A real matrix $\boldsymbol{\mathcal{A}}$
$\boldsymbol{\mathcal{A}}^T$	A real matrix $\boldsymbol{\mathcal{A}}$ transposed
$\boldsymbol{\mathcal{A}}^{-T}$	Matrix $\boldsymbol{\mathcal{A}}$ inverted and transposed
$\mathbb{R}^N$	The set of real vectors of size N
$\mathbb{R}^{N\times N}$	The set of real matrices of size $N \times N$
$\mathbb{C}^N$	The set of complex vectors of size N
$\mathbb{C}^{N\times N}$	The set of complex matrices of size $N \times N$
$d_{i,j}$	Element $d_{i,j}$ in i^{th} row and j^{th} column of a matrix
$\boldsymbol{I}(x,s)$	Laplace-domain current at distance x for a distributed network.
$\boldsymbol{V}(x,s)$	Laplace-domain voltage at distance x for a distributed network.
$\boldsymbol{\mathcal{U}}$	The identity matrix
$\boldsymbol{\mathcal{U}}_{m\times m}$	An $m \times m$ identity matrix.
$\boldsymbol{\mathcal{I}}_m$	An $m \times m$ identity matrix.

Chapter 1

Introduction

1.1 Motivation

The recent developments in electronics devices in terms of features offered and portability of equipments have highlighted the importance of *Electronics Design Automation* (EDA) tools [1]. Since the 1980's, EDA tools have been recognized as a main design aid in managing complex circuit designs. Ever since, the inclusion of EDA tools in all design phases has drastically reduced the time consumed by each phase. However, the relation is mutually reversible in the sense that the advances in EDA tools have made complex designs realizable. For example the notion of having a *System-on-Chip* (SoC) is now a realty due to recent advances in submicron transistor packaging and integration. SoC is a typical example of complex structures where mixed-mode operation of analog and digital circuits coexist. The integration of analog and digital circuits into single chip increases the size of the equivalent mathematical representation of the original circuit. Along with the increase in packaging density, there is also an increase in operating frequencies. The increase in operating frequencies has a strong impact on the wires that connect different modules. These wires can no longer be modeled as simple short circuits with negligible effects. More precise models such as lossy Multi-conductor

Transmission Lines (MTL) [1] are incorporated to describe the effect of these wires on signal propagation. Clearly, having an MTL increases the simulation problem drastically.

New process technologies typically require more complex models to represent various physical phenomena at the nanoscale levels. These technologies introduce new models for functional electronics. The incorporation of these models into the design phase further enlarges the problem at hand.

The concept of Model-Order Reduction (MOR) has been introduced [2] to address the issue of increasing computational complexity that arises in simulating the response of modern circuit designs. The basic idea in MOR techniques relies on constructing an approximate system for the original system, but with much smaller size. MOR has been very successful in different application domains, e.g. Finite-Difference Time-Domain (FDTD) and Finite-Elements Methods (FEM) [3], [4].

In addition to simulating the circuit response, computing the sensitivity of such response with respect to one or more design parameters is a vital requirement during the design phase. The availability of sensitivity information enables effective techniques to be used to compute the impact of process variability on design performance, carry out statistical and yield analysis, and perform automatic design optimization. For example, gradient-based techniques mandate that the sensitivity information be known w.r.t. every design parameter such that local enhancement to current design configuration could be effectively computed. A classical and efficient way to compute sensitivity is based on *adjoint network* concept [5]. Using the concept of adjoint network enables computing the sensitivity of circuit response w.r.t. all designable parameters using only two system analyses regardless of the number of these parameters.

1.2 Objective of the Thesis

The objective of this thesis is to enable the application of time-domain sensitivity analysis, based on the concept of adjoint networks, to reduced-order systems constructed through MOR techniques. Such an application would provide means to combine the computational efficiency of MOR techniques with the computational convenience offered by the concept of adjoint networks. A brief description of the contribution of this thesis in this regard is outlined in the next section.

1.3 Contributions

The following are the main contributions presented in the thesis [6], [7].

1. The derivation of a new algorithm to compute the sensitivity of circuits having large linear systems as subnetworks, where those subnetworks are being represented by reduced-order models obtained via one of the common MOR techniques . The large subnetworks are assumed to arise from finite discretization of some microwave device and are represented by a large number of lumped components. The interconnection between these subnetworks are assumed to be a set of lumped linear or nonlinear elements, and the sensitivity information required is the sensitivity of the performance metrics of the entire circuit w.r.t. the designable parameters of the subnetworks.

2. Implementing the sensitivity analysis within the framework of adjoint principal, where only two time-domain analyses, one for the original system and another for the adjoint network, need to be performed to obtain the sensitivity information w.r.t. all designable parameters, regardless of the number of those parameters

3. Showing that the proposed algorithm can be implemented on top of existing MOR tech-

niques with minimal increment in computational cost. This facilitates using intermediate terms generated in the course of MOR to be employed towards sensitivity computation and enables existing code of MOR to be reused for sensitivity analysis.

1.4 Background

This section provides brief background for the main topics needed for the presentation of this work. It also provides the appropriate context to describe the proposed scheme.

1.4.1 Linear Circuits

Large linear circuits are intrinsic to current complex and high frequency designs. They arise in modeling techniques that rely on gridding. For instance, a descretization of Telegrapher's equations that describes MTL along its spatial axis would result in large number of equivalent **RLC** network. This technique has been used to simulate MTL up to certain operating frequency. However, the resultant equivalent network introduces many extra nodes and therefore increases the problem size significantly.

Another example where large linear circuits are needed, appears in Partial-Element Equivalent-Circuit (PEEC) technique, where wave propagation in a guided medium is studied by spatially dividing that medium into equivalent **RLC** grids. This gridding is done in 3-d approach where the grid size depends on the shortest wavelength the medium is going to carry. It is suggested that the grid unit to be of 0.1 of the shortest wavelength. This results in a very large linear system.

Other techniques such as Finite-Difference Time-Domain (FDTD) and Finite-Element Methods (FEM) result in huge linear systems as well [8], [9].

1.4.2 Model-Order Reduction (MOR)

The concept of MOR has been used, with reported success, over the past decade to address the high computational cost arising from having to deal with large linear circuits in the simulation phase. More specifically, MOR based on *Krylov subspace methods* have been well recongnized[1] in the literature due to their efficiency in approximating large scale systems. The basic idea in these techniques is to find a bidirectional mapping operator that maps the high-dimensional system into an equivalent and reduced-dimension system and vice versa [11], [12], [13], [14], [15]. The responses of both systems for a given input are almost identical. The basic objective of the projection operator is to preserve the q leading eigenvalues of the original system. To further illustrate the underlying concept, consider an N-dimentional system that has m input ports and n output ports and therefore can be represented by the following N-dimensional state space equations:

$$\frac{d\boldsymbol{x}(t)}{dt} = \boldsymbol{\mathcal{A}}\boldsymbol{x}(t) + \boldsymbol{\mathcal{B}}\boldsymbol{u}(t) \tag{1.1}$$

$$\boldsymbol{y}(t) = \boldsymbol{\mathcal{C}}^T\boldsymbol{x}(t) + \boldsymbol{\mathcal{D}}\boldsymbol{u}(t) \tag{1.2}$$

where $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{N\times N}$, $\boldsymbol{\mathcal{B}} \in \mathbb{R}^{N\times m}$, $\boldsymbol{\mathcal{C}} \in \mathbb{R}^{N\times n}$ and $\boldsymbol{\mathcal{D}} \in \mathbb{R}^{n\times m}$.

A Krylov subspace MOR typically computes the projector, which in this context is denoted by a matrix $\boldsymbol{\mathcal{Q}} \in \mathbb{R}^{N\times q}$. This matrix is then used to reduce the original system (1.1) and (1.2) into the following system:

$$\frac{d\tilde{\boldsymbol{x}}(t)}{dt} = \tilde{\boldsymbol{\mathcal{A}}}\tilde{\boldsymbol{x}}(t) + \tilde{\boldsymbol{\mathcal{B}}}\boldsymbol{u}(t) \tag{1.3}$$

$$\tilde{\boldsymbol{y}}(t) = \tilde{\boldsymbol{\mathcal{C}}}^T\tilde{\boldsymbol{x}}(t) + \boldsymbol{\mathcal{D}}\boldsymbol{u}(t) \tag{1.4}$$

[1] Krylov subspace iteration methods were mentioned among the top 10 algorithms of the 20$^{\text{th}}$ century [10].

where $\tilde{\boldsymbol{\mathcal{A}}} \in \mathbb{R}^{q\times q}$, $\tilde{\boldsymbol{\mathcal{B}}} \in \mathbb{R}^{q\times m}$, and $\tilde{\boldsymbol{\mathcal{C}}} \in \mathbb{R}^{q\times n}$.

It is clear that if $q \ll N$, which is usually the case, the size of the reduced system (1.3)-(1.4) becomes much smaller than the size of the original system. The equivalence between the reduced and the original systems can be demonstrated through showing that they share the first q block moments. It is also beneficial to simulate the approximate system where the system size q is very small compared to the original system size.

The approaches of generating of the mapping operator (projection operator) could be subdivided into two categories; explicit moment matching techniques and implicit moment matching technique. The explicit moment matching technique suffers numerical deterioration for relatively small q whereas generation of the mapping operator under the latter techniques is known to numerically perform better. It has also been shown that the latter techniques can maintain the passivity of the original model; a desirable property that guarantees the stability of time-domain simulation.

1.4.3 Sensitivity Analysis

The term sensitivity analysis in the context of this thesis refers to computing the derivatives of circuit performance metrics w.r.t. the circuit designable parameters, i.e. those parameters that can be varied within certain specified constraints in order to attain optimum performance metrics. An example of performance metric is the rise/fall time of a digital pulse or the bandwidth of a microwave filter, whereas the width of a conductor or the sizing of an integrated transistor could serve as examples of design parameters. Obtaining the sensitivity information directly in the time domain is a key design-aid in the design cycle of general nonlinear circuits, since nonlinear elements can be represented only in the time domain [16], [17], [18], [19]. There are two principal approaches to computing the sensitivity in the time domain, namely those based on the direct differentiation and these based on adjoint-based techniques. An approach was

described in [20] to combine the advantages of the two techniques.

Adjoint Sensitivity

The numerical advantage of adjoint sensitivity stems from the fact that only two system solutions are needed to obtain the sensitivity information, of a single waveform, for various design parameters [21], [22], [23], [24]. Further, adjoint sensitivity information could be computed in time domain and frequency domain. However, the latter case is restricted to pure linear circuits since nonlinear elements can be described in the time domain only. Therefore, time-domain adjoint sensitivity covers broader class of problems. In the frequency domain, sensitivity information could be selectively obtained for any frequency point of interest. However, in time-domain sensitivity, this information is obtained from the solution of two systems (the original and adjoint) through forward and reverse time stepping with the proper initial and terminal conditions, respectively.

Direct-Differentiation Sensitivity

In contrast to adjoint sensitivity analysis, direct differentiation technique yields the sensitivity of *all* circuit variables w.r.t. a *single* designable parameter via a single system analysis. Here, the system matrices are those that describes the original circuit, while the stimulus vector of sources is obtained from the derivatives of those matrices w.r.t. the designable parameter and the solution of the circuit waveforms [25]. Hence, computing the sensitivity of a circuit with, say N, designable parameters using direct differentiation requires running N systems analyses with the same matrices of the original circuit, but with N different stimulus vectors. Therefore, adjoint based sensitivity analysis is preferred in situations where one needs the sensitivity of few variables with respect to a large number of design parameters.

1.5 Thesis Organization

Chapter 2 provides necessary information about projection-based model-order reduction techniques. The merits of the model-order reduction techniques stemming from controllability and observability of control theory are presented. In Chapter 3, we present the current efficient techniques that are widely used to compute sensitivity information of electrical networks. Chapter 4 investigates the problem of combining model-order reduction with adjoint-based sensitivity in the time domain and examines the main difficulty that hinders such a combination. Chapter 5, which presents the main contribution of this thesis, develops an algorithm that facilitates this combination. Numerical results for practical design issue and for exercising the proposed technique in different domains are presented in Chapter 6. Chapter 7 concludes this work and present possible future work.

Chapter 2

Model-Order Reduction

This chapter lays out the basic theoretical foundation behind the concept of the Model-Order Reduction (MOR) techniques. To facilitate this presentation of MOR, the chapter starts first in Section 2.1 by describing the standard formulation for general network that have both linear and nonlinear elements.

2.1 Modified Nodal Analysis Formulation

The task of estimating the response of general nonlinear circuits requires a formulation approach that can be applied, in an automated manner, to describe the circuit in the form of a set of differential equations. One of the approaches that have been widely adopted due to their efficiency is based on the idea of Modified Nodal Analysis (MNA)[1]. MNA provides a systematic way to construct the network constituent equations without resorting to the well-known Kirchoff's current and voltage laws for each circuit element. Each electrical element of m ports is ideally connected to m nodes of a given network, and hence a node[2] is defined as a net-

[1]Some authors refer to the MNA as Modified Nodal Admittance. However, there are changes made to the admittance matrix that make the former name more suitable

[2]A node could be defined as a network point where three or more electrical elements are connected. However, this will require special handling and raise many special cases

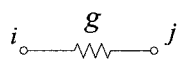
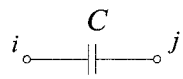
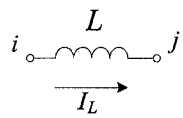
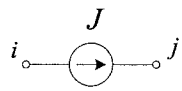
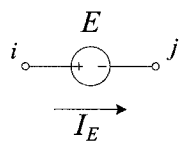
Element	Circuit Schematic	Stamp
Resistor		$\begin{array}{c} \\ i \\ \\ j \end{array} \begin{array}{c} \begin{array}{ccc} i & \cdots & j \end{array} \\ \begin{bmatrix} g & \cdots & -g \\ \vdots & \ddots & \vdots \\ -g & \cdots & g \end{bmatrix} \end{array}$
Capacitor		$\begin{array}{c} \\ i \\ \\ j \end{array} \begin{array}{c} \begin{array}{ccc} i & \cdots & j \end{array} \\ \begin{bmatrix} C & \cdots & -C \\ \vdots & \ddots & \vdots \\ -C & \cdots & C \end{bmatrix} \end{array}$
Inductor		$\begin{array}{c} \\ i \\ \\ j \\ I_L \end{array} \begin{array}{c} \begin{array}{cccc} i & \cdots & j & I_L \end{array} \\ \begin{bmatrix} 0 & \cdots & 0 & 1 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & -1 \\ 1 & \cdots & -1 & -sL \end{bmatrix} \end{array}$
Current Source		$\begin{array}{c} \\ i \\ \\ j \end{array} \begin{bmatrix} J \\ \vdots \\ -J \end{bmatrix}$
Voltage Source		$\begin{array}{c} \\ i \\ \\ j \\ I_E \end{array} \begin{array}{c} \begin{array}{cccc} i & \cdots & j & I_E \end{array} \\ \begin{bmatrix} 0 & \cdots & 0 & 1 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & -1 \\ 1 & \cdots & -1 & 0 \end{bmatrix} \end{array} \begin{bmatrix} \\ \\ \\ E \end{bmatrix}$

Figure 2.1: Few basic circuit elements and their stencils

work point where two or more electrical elements are connected. Each electrical element of m ports is assigned a stamp (stencil) which defines how this element would affect various node voltages and/or branch currents. Figure 2.1 presents stencil information for basic electrical elements [5].

2.1.1 Linear Networks

Linear networks with frequency-independent elements take on the following form of equations:

$$\begin{aligned} \boldsymbol{\mathcal{C}}\dot{\boldsymbol{x}}(t) &= -\boldsymbol{\mathcal{G}}\boldsymbol{x}(t) + \boldsymbol{\mathcal{B}}\boldsymbol{u}(t) \\ \boldsymbol{y}(t) &= \boldsymbol{\mathcal{L}}^T\boldsymbol{x}(t) + \boldsymbol{\mathcal{D}}\boldsymbol{u}(t) \end{aligned} \tag{2.1}$$

where $\boldsymbol{x}(t) \in \mathbb{R}^N$ is a vector of network unknown node voltages and branch currents of inductors and voltage sources. $\boldsymbol{\mathcal{G}}$ and $\boldsymbol{\mathcal{C}} \in \mathbb{R}^{N\times N}$ are constant matrices and contain stamps of the memoryless and memory elements, respectively. $\boldsymbol{u}(t) \in \mathbb{R}^m$ contains waveform information of independent current sources and voltage sources. Matrices $\boldsymbol{\mathcal{B}}$, $\boldsymbol{\mathcal{L}}$, and $\boldsymbol{\mathcal{D}}$ are in $\mathbb{R}^{N\times m}$, $\mathbb{R}^{N\times n}$, and $\mathbb{R}^{n\times m}$ respectively.

The system in (2.1) could be rewritten, assuming zero initial conditions, in frequency domain as:

$$\begin{aligned} s\boldsymbol{\mathcal{C}}\boldsymbol{X}(s) &= -\boldsymbol{\mathcal{G}}\boldsymbol{X}(s) + \boldsymbol{\mathcal{B}}\boldsymbol{U}(s) \\ \boldsymbol{Y}(s) &= \boldsymbol{\mathcal{L}}^T\boldsymbol{X}(s) + \boldsymbol{\mathcal{D}}\boldsymbol{U}(s) \end{aligned} \tag{2.2}$$

Several techniques have been employed to solve the system of equations defined in (2.1). Those techniques make use of numerically computed derivative information to extrapolate the value of the next time point in the vicinity of the current point of time. In electronic circuits, it is not unusual to have $\boldsymbol{\mathcal{C}}$ to be a sparse matrix leading to a simultaneous sets of algebraic and differential equations known as Differential Algebraic Equations (DAE). However, numerical integration of DAE's suffers from instability and inaccuracy issues. An integration formulae is stable if no errors accumulates over time and is accurate if local truncation error is low.

Solution of frequency-domain set of equations defined in (2.2) is straightforward matter. In fact, SPICE-like programs make use of the fact that the system matrix $(\boldsymbol{\mathcal{G}} + s\boldsymbol{\mathcal{C}})$ is sparse to re-

duce the computation burden at every frequency point of the solution. Normally, Lower/Upper decomposition (L/U) is performed on the system matrix for a given frequency point. The solution is found by performing a forward and a backward substitutions. The complexity order of L/D is of $\Omega(N^{\alpha})$, and $\alpha = 1.1 - 1.5$ depending on the sparsity pattern of the system matrix. This is a better performance compared to matrix inversion which is of order $\Omega(N^3)$. The order of forward/backward substitution is $\Omega(N^2)$. Another category exists that computes the solution for these systems iteratively without matrix decomposition [26]. Those iterative techniques are memory-wise efficient and produce relatively accurate solutions. The cost of matrix-vector multiplication for dense matrices is comparable to the cost of a forward/backward substitutions and is much smaller in case of sparse matrices. However, the number of iterations to reach a solution point is variable and depends on the stiffness of the problem at hand.

2.1.2 Nonlinear Networks

Modified Nodal Analysis can also describe nonlinear networks where the vector of unknowns is appended by extra variables that describe the current/charge/flux flowing through nonlinear elements. The network equation takes on the following form:

$$\begin{aligned} \boldsymbol{C}\dot{\boldsymbol{x}}(t) &= -\boldsymbol{\mathcal{G}}\boldsymbol{x}(t) + \boldsymbol{\mathcal{B}}\boldsymbol{u}(t) - \boldsymbol{f}(\boldsymbol{x}(t)) \\ \boldsymbol{y}(t) &= \boldsymbol{\mathcal{L}}^T\boldsymbol{x}(t) + \boldsymbol{\mathcal{D}}\boldsymbol{u}(t) \end{aligned} \tag{2.3}$$

where $\boldsymbol{f}(\boldsymbol{x}(t))$ is a vector of functions that describes various nonlinear relations of nonlinear elements.

Nonlinear networks could be described as given in (2.3) in time domain only. Integration formula need to be applied as in their counterparts define in (2.1). However, at every time point, computation of the solution at the current time point involves solving the nonlinear

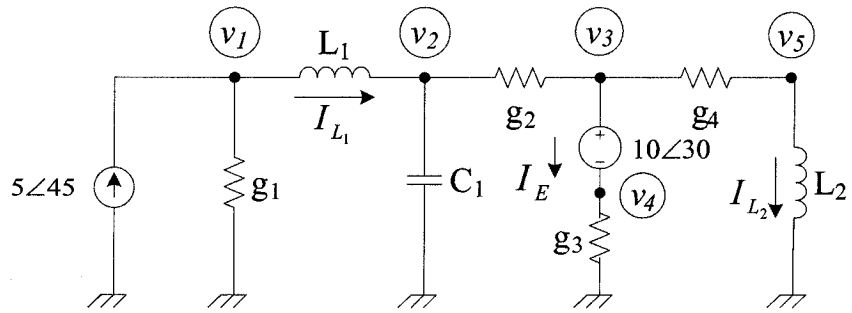


Figure 2.2: Linear circuit with basic electrical components

equations along with the the integral equations. Newton-Raphson (NR) techniques are widely used to solve the nonlinear network at every time point. For ideal cases, NR has a quadratic convergence criterion which makes it approach the solution point in few number of steps. However, if the initial (guess) point is not well chosen or the function has a zero of order higher than one, NR will fail to reach the solution point. In the first case, it will suffer from indefinite oscillations and in the latter case it will abort due to a divide-by-zero error. A remedy exists to avoid the divide-by-zero case where the function divided by its first derivative replaces the original function. This remedy is easily perceivable for one dimension function as the zero order is reduced to one.

Example 1

Figure 2.2 shows a linear circuit with some basic electrical components. This circuit is described using MNA formulation as follows:

$$\boldsymbol{\mathcal{G}}\boldsymbol{x}(t) + \boldsymbol{C}\dot{\boldsymbol{x}}(t) = \boldsymbol{\mathcal{B}}\boldsymbol{u}(t)$$

where the constituent matrices take on the following formats:

$$\boldsymbol{\mathcal{G}} = \begin{bmatrix} g_1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & g_2 & -g_2 & 0 & 0 & -1 & 0 & 0 \\ 0 & -g_2 & g_2+g_4 & 0 & -g_4 & 0 & 0 & 1 \\ 0 & 0 & 0 & g_3 & 0 & 0 & 0 & -1 \\ 0 & 0 & -g_4 & 0 & g_4 & 0 & 1 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\boldsymbol{\mathcal{C}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & C_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -L_1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -L_2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\boldsymbol{\mathcal{B}}(t) = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}, \boldsymbol{u}(t) = \begin{bmatrix} 5cos(\omega t + 45) \\ 10cos(\omega t + 30) \end{bmatrix}, \text{ and } \boldsymbol{x}(t) = \begin{bmatrix} v_1(t) \\ v_2(t) \\ v_3(t) \\ v_4(t) \\ v_5(t) \\ I_{L_1}(t) \\ I_{L_2}(t) \\ I_E(t) \end{bmatrix}$$

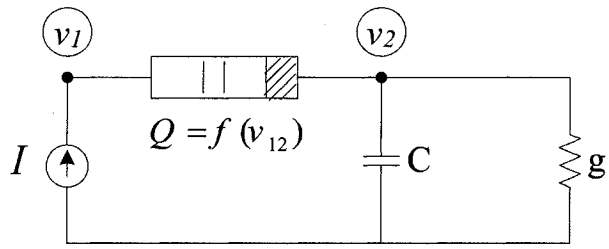


Figure 2.3: Nolinear electrical circuit

Example 2

Equations governing nonlinear electrical elements could be augmented into MNA formulation. Figure 2.3 shows a simple circuit with nonlinear capacitor. This circuit could be described using MNA formulation with constituent matrices that look as follow:

$$\mathcal{G} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & g & 0 \\ 0 & 0 & -1 \end{bmatrix}, \mathcal{C} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & C & -1 \\ 0 & 0 & 0 \end{bmatrix}, \mathcal{B}(t) = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix},$$

$$\boldsymbol{u}(t) = \begin{bmatrix} I \end{bmatrix}, \boldsymbol{x}(t) = \begin{bmatrix} v_1(t) \\ v_2(t) \\ Q(t) \end{bmatrix}, \text{and } \boldsymbol{f}(\boldsymbol{x}(t)) = \begin{bmatrix} 0 \\ 0 \\ f(v_1(t) - v_2(t)) \end{bmatrix}$$

2.2 Model-Order Reduction for Linear Networks

Model-Order Reduction finds an approximate system that behaves the same as the original system in the proximity of an expansion point in the frequency domain. The idea is simple; although system transfer function has many driving moments, only few of them have the substantial effect on the output waveforms. Abstracting original system to another reduced-order system that has only these dominant moments would result in accountable error [27], [28]. This error could be perceived up to certain simulation time/frequency. As will be seen in the next subsections, increasing the simulation time or frequency would increase the error asso-

ciated with the reduced-order system. Algorithms that find the approximate system could be divided into two categories; explicit moment matching and implicit moment matching.

2.2.1 Explicit Moment Matching

Explicit Moment Matching (EMM) expands the system[3] transfer function at the vicinity of a desired frequency point. Consider, for example, the single-input single-output (SISO) system transfer function represented by its Taylor series expansion as:

$$h(s) = m_0 + m_1(s - s_\circ) + m_2(s - s_\circ)^2 + \cdots m_l(s - s_\circ)^l + \cdots \tag{2.4}$$

where s is the complex frequency, $m_i = \frac{1}{i!} \frac{dh(s)}{ds}\Big|_{s=s_\circ}$ is the i^{th} moment (scaled derivative), and $s_\circ$ is the expansion point. However, the above expansion (2.4) does not capture the highly nonlinear nature of the transfer function. It was suggested that Padé approximation, with its pole-capture capability, would be a better representation of the system response. In Padé approximation, system transfer function is to be approximated by a rational function of the form:

$$P(L, M) = \frac{N_L(s)}{D_M(s)} \tag{2.5}$$

where $N_L(s)$ and $D_M(s)$ are polynomial functions (of s) of order L, M respectively. L and M are free variables and are chosen relevant to problem at hand.

It is desired that frequency response of the original system defined in (2.4) matches the frequency response of system defined in (2.5). This can be formulated as the following system

[3]Single-Input Single-Output (SISO) systems are meant here or in general case port-to-port transfer function would be used

of equations:

$$\begin{aligned} h(s) &= \frac{a_0 + a_1(s-s_\circ) + a_2(s-s_\circ)^2 + \cdots + a_L(s-s_\circ)^L}{1 + b_1(s-s_\circ) + b_2(s-s_\circ)^2 + \cdots + b_M(s-s_\circ)^M} \\ &= m_0 + m_1(s-s_\circ) + m_2(s-s_\circ)^2 + \cdots + m_{L+M}(s-s_\circ)^{L+M} \end{aligned} \tag{2.6}$$

The solution of the above system could be evaluated by cross-multiplying opposite sides and grouping coefficients of similar powers of s. Solving for these coefficients would result in the following system:

$$\begin{bmatrix} m_{L-M+1} & m_{L-M+2} & \cdots & m_L \\ m_{L-M+2} & m_{L-M+3} & \cdots & m_{L+1} \\ \vdots & \vdots & \ddots & \vdots \\ m_L & m_{L+1} & \cdots & m_{L+M-1} \end{bmatrix} \begin{bmatrix} b_M \\ b_{M-1} \\ \vdots \\ b_1 \end{bmatrix} = - \begin{bmatrix} m_{L+1} \\ m_{L+2} \\ \vdots \\ m_{L+M} \end{bmatrix} \tag{2.7}$$

After solving systems of Equations (2.7), b_i coefficients will be known, a_i could be obtained as follows:

$$\begin{aligned} a_0 &= m_0 \\ a_1 &= m_1 + b_1 m_0 \\ a_2 &= m_2 + b_1 m_1 + b_2 m_0 \\ &\vdots \\ a_i &= m_i + \sum_{j=1}^{\min(i,M)} b_j m_{i-j} \end{aligned} \tag{2.8}$$

where $i = 0, 1, 2, \ldots, L$.

The process of finding a_i's and b_i's of SISO system transfer function is of order $\Omega(\max(M^3, L^2))$. However, for multi-port network of multiple transfer functions, the order of finding a Padé ap-

proximation is equivalent to finding the DC solution point of the original system [1] which remains a low overhead.

Identification of m_i's is straightforward. Consider the linear and time-invariant system of equations defined in (2.2) whose system transfer function could be devised to be:

$$\boldsymbol{Y}(s) = \boldsymbol{\mathcal{L}}^T(\boldsymbol{\mathcal{G}} + s\boldsymbol{\mathcal{C}})^{-1}\boldsymbol{\mathcal{B}} \tag{2.9}$$

Expansion of (2.9) using Taylor expansion around $s_\circ$ and comparing similar power of s would yield m_i's as follows:

$$\begin{aligned} \boldsymbol{Y}(s) &= \boldsymbol{\mathcal{L}}^T(\boldsymbol{\mathcal{I}}_N + s\underbrace{\boldsymbol{\mathcal{G}}^{-1}\boldsymbol{\mathcal{C}}}_{\boldsymbol{\mathcal{A}}})^{-1}\underbrace{\boldsymbol{\mathcal{G}}^{-1}\boldsymbol{\mathcal{B}}}_{\boldsymbol{\mathcal{R}}} \\ &= \boldsymbol{\mathcal{L}}^T(\boldsymbol{\mathcal{R}} + s\boldsymbol{\mathcal{A}}\boldsymbol{\mathcal{R}} + s^2\boldsymbol{\mathcal{A}}^2\boldsymbol{\mathcal{R}} + \ldots + s^i\boldsymbol{\mathcal{A}}^i\boldsymbol{\mathcal{R}} + \ldots) \end{aligned} \tag{2.10}$$

and hence,

$$\begin{aligned} \boldsymbol{\mathcal{M}}_0 &= \boldsymbol{\mathcal{R}} \\ \boldsymbol{\mathcal{M}}_1 &= \boldsymbol{\mathcal{A}}\boldsymbol{\mathcal{M}}_0 \\ &\vdots \\ \boldsymbol{\mathcal{M}}_i &= \boldsymbol{\mathcal{A}}^i\boldsymbol{\mathcal{M}}_0 \end{aligned} \tag{2.11}$$

where $\boldsymbol{\mathcal{M}}_i$ represents the i^{th} block moment of the network.

One of the major drawbacks of EMM is seen in Equation (2.11) where the system matrix $\boldsymbol{\mathcal{A}}$ is raised to power i. As the power increases, the resultant matrix $\boldsymbol{\mathcal{A}}^i$ becomes ill-conditioned and the vector space spanned becomes a single line coinciding with the eigenvector of the largest absolute eigenvalue. Experiments had shown that matrix $\boldsymbol{\mathcal{A}}^i$ becomes ill-conditioned for $i > 10$ [1]. A remedy that cleverly addresses this issue is CFH algorithm where multiple expansion point are presumed to approximate the original system [29], [30].

Another problem of EMM is that it can not easily be incorporated into time-domain representation. For complex systems where m-to-n transfer functions exist, it is desirable to have common pole set for off-diagonal transfer functions. This problem is addressed in [31], [32], where common pole set is defined for off-diagonal transfer functions.

Another problem that EMM suffers from is that Padé approximation by definition does not warrant the passivity of the original passive network. Loss of passivity can produce unstability in the reduced-order model when it is connected with other systems leading to numerically oscillating waveforms. This problem hinders the use of EMM as defined above.

2.2.2 Implicit Moment Matching

Implicit Moment Matching (IMM) techniques have gained more interest due to their inherent abilities to overcome shortcomings of EMM techniques. The basic idea of IMM techniques relies on eliminating the uncontrollable and unobservable states of system transfer functions. We consider the SISO system presented earlier and whose transfer function was described by (2.4) as a way of illustrating the underlying concept behind IMM. It can be shown that such a transfer function can be expressed in the following diagonal form:

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{\mathcal{T}}^{-1}\boldsymbol{\mathcal{A}}\boldsymbol{\mathcal{T}}\boldsymbol{x} + \boldsymbol{\mathcal{T}}\boldsymbol{B}u = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} \hat{b}_1 \\ \vdots \\ \hat{b}_n \end{bmatrix} u \\ y &= \boldsymbol{L}^T\boldsymbol{\mathcal{T}}\boldsymbol{x} = \begin{bmatrix} \hat{l}_1 & \cdots & \hat{l}_n \end{bmatrix} x \end{aligned} \tag{2.12}$$

where $\boldsymbol{\mathcal{T}}$ is the matrix of eigenvectors of the system matrix. Figure 2.4 shows a block diagram realization for the system defined in (2.12).

Kalman's controllability and observability principles showed that uncontrollable and un-

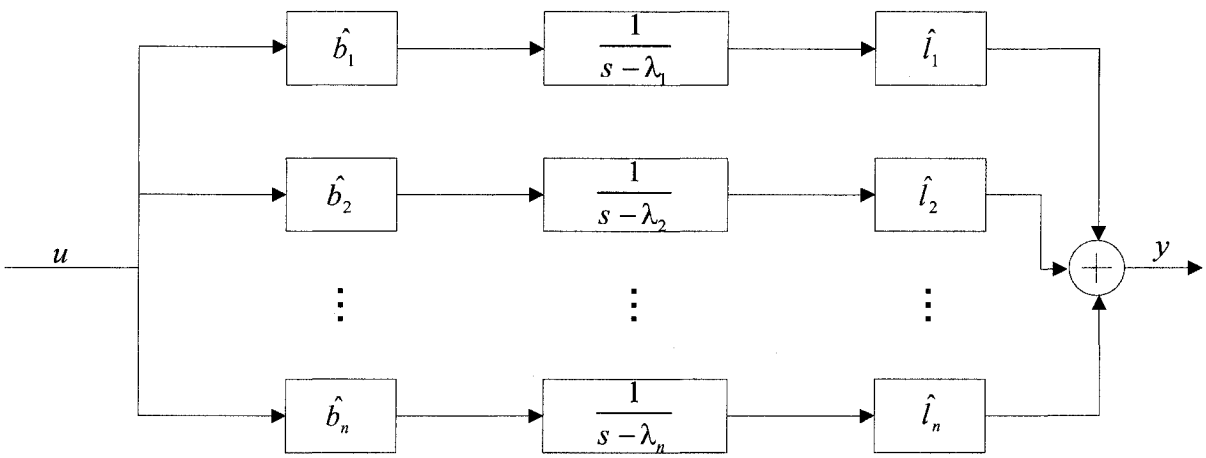


Figure 2.4: Block diagram of SISO system defined in (2.12)

observable states could be eliminated without affecting the system transfer function. A state i is uncontrollable when $\hat{b}_i$ is equal to zero and is unobservable when $\hat{l}_i$ is equal to zero. In addition to those states that are completely uncontrollable or unobservable,"weakly" controllable [4] and weakly observable [5] states can also be eliminated without compromising the accuracy of the original system. The key idea in model-order reduction is to identify those uncontrollable, unobservable, weakly controllable, or weakly observable states and eliminate those states.

One way for the elimination of [weakly] uncontrollable and [weakly] unobservable states could be achieved by the following transformation:

$$\begin{aligned} \dot{\hat{\boldsymbol{x}}} &= \underbrace{\boldsymbol{\mathcal{T}}_{upper}^{-1}\boldsymbol{A}\boldsymbol{\mathcal{T}}_{left}}_{\hat{\boldsymbol{A}}}\hat{\boldsymbol{x}} + \underbrace{\boldsymbol{\mathcal{T}}_{upper}^{-1}\boldsymbol{B}}_{\hat{\boldsymbol{B}}}u \\ \boldsymbol{y} &= \underbrace{\boldsymbol{L}^T\boldsymbol{\mathcal{T}}_{left}}_{\hat{\boldsymbol{L}}^T}\hat{\boldsymbol{x}} \end{aligned} \tag{2.13}$$

where $\boldsymbol{\mathcal{T}}_{left}$ and $\boldsymbol{\mathcal{T}}_{upper}^{-1}$ are rectangular matrices with sizes $(n \times q)$ and $(q \times n)$, respectively.

[4] state i is weakly controllable iff $\hat{b}_i$ is negligibly small

[5] similarly, state i is weakly observable iff $\hat{l}_i$ is negligibly small

It should be obvious by now that the size of the reduced system is q, and the MOR algorithm is successful if it can retain the q most controllable and observable states, while $q \ll n$.

To achieve that goal the column space of the matrix $\mathcal{T}_{left}$ must span the column space of the q leading eigenvectors of the system matrix[6] $\mathcal{A}$ i.e. the first q columns in the matrix $\mathcal{T}$ whereas the rows in $\mathcal{T}^{-1}_{upper}$ should span the corresponding first q rows of the $\mathcal{T}^{-1}$. Vector $\hat{\boldsymbol{x}} \in \mathbb{R}^{q\times 1}$ represents those states corresponding to the most controllable and observable states.

The above operation mathematically defines a projection operation where the original system is projected onto a space (spaces) [7] defined by $\mathcal{T}_{left}$ (and $\mathcal{T}^{-1}_{upper}$). Computational burden here reduces to computing only $2q$ vectors of dimension n, the columns of $\mathcal{T}_{left}$ and rows of $\mathcal{T}^{-1}_{upper}$. This is in essence a significant saving over the system defined in (2.12) as the whole matrix $\mathcal{T}$ need not be computed. It is worth noting that matrices $\mathcal{T}_{left}$ and $\mathcal{T}^{-1}_{upper}$ are biorthogonal matrices ($\mathcal{T}_{left} \times \mathcal{T}^{-1}_{upper} = \mathcal{I}$).

Central to the issues related to MOR is the procedure used to compute the transformation operators $\mathcal{T}_{left}$ and $\mathcal{T}^{-1}_{upper}$. In the context of systems that arise from large linear circuits, *Krylov* subspace methods have been widely adopted due to their computational efficiency. Recall that for linear circuits, a system is usually described using MNA formulation as shown in (2.2). If we denote the left and right operator matrices by $\mathcal{P}$ and $\mathcal{Q}$, the reduced system in that case can be written as follows:

$$\begin{aligned} \underbrace{\mathcal{P}^T\mathcal{C}\mathcal{Q}}_{\hat{\mathcal{C}}}\dot{\hat{\boldsymbol{x}}} &= \underbrace{\mathcal{P}^T\mathcal{G}\mathcal{Q}}_{\hat{\mathcal{G}}}\hat{\boldsymbol{x}} + \underbrace{\mathcal{P}^T\boldsymbol{B}}_{\hat{\boldsymbol{B}}}u \\ y &= \underbrace{\boldsymbol{L}^T\mathcal{Q}}_{\hat{\boldsymbol{L}}^T}\hat{\boldsymbol{x}} \end{aligned} \tag{2.14}$$

where $\mathcal{P}$ and $\mathcal{Q}$ are projection operators of size $N \times q$, with typically $q \ll N$.

[6]The leading eigenvectors here refer to those eigenvectors associated with the eigenvalues having the largest absolute values

[7]as will be seen later, two different matrices could be used in the projection process

The matrices $\mathcal{P}$ and $\mathcal{Q}$ used to reduce the circuit are chosen as orthogonal bases for the Krylov subspaces defined by the system matrices. Generally, a Krylov subspace associated with a matrix $\boldsymbol{A}$ and vector $\boldsymbol{B}$ and order q is defined as:

$$\mathrm{K}_q(\boldsymbol{A}, \boldsymbol{r}) = \mathrm{span}\{\boldsymbol{r}, \boldsymbol{A}\boldsymbol{r}, \boldsymbol{A}^2\boldsymbol{r}, \ldots, \boldsymbol{A}^{q-1}\boldsymbol{r}\} \tag{2.15}$$

where $\boldsymbol{A}$ is a constant matrix of size $N \times N$, $\boldsymbol{r}$ is a constant vector of size N, and q, an integer greater than zero, defines the Krylov subspace dimension.

In this context $\mathcal{P}$ and $\mathcal{Q}$ are taken as bases for the following Krylov subspaces:

$$\mathcal{Q} = \text{Column Span}[\mathrm{K}_q(\boldsymbol{A}_Q, \boldsymbol{r}_Q)] \tag{2.16}$$

$$\mathcal{P} = \text{Column Span}[\mathrm{K}_q(\boldsymbol{A}_P, \boldsymbol{r}_P)] \tag{2.17}$$

where $\boldsymbol{A}_Q = \boldsymbol{G}^{-1}\boldsymbol{C}$, $\boldsymbol{A}_P = \boldsymbol{G}^{-T}\boldsymbol{C}^T$, $\boldsymbol{r}_Q = \boldsymbol{G}^{-1}\boldsymbol{B}$, and $\boldsymbol{r}_P = \boldsymbol{G}^{-T}\boldsymbol{L}$.

In addition to preserving the q most controllable and observable states in the system, this particular choice for $\mathcal{P}$ and $\mathcal{Q}$ preserves at least the first q moments of the original system. This property is crucial in guaranteeing the accuracy of subsequent circuit simulation since it makes the frequency domain behavior of the reduced system almost indistinguishable from that of the original system.

More precisely, if we use $\hat{m}_i$ to denote the i^{th} moment of the transfer function of the reduced system (1.3) and (1.4) while we let m_i be the moment of the transfer function of the original system, then it can be shown that the sufficient condition to have

$$\hat{m}_i = m_i \tag{2.18}$$

where $0 < i < k$ for some k, is that $\mathcal{Q}$ and $\mathcal{P}$ span the Krylov subspaces $\mathrm{K}_q(\boldsymbol{A}_Q, \boldsymbol{r}_Q)$ and

$\mathrm{K}_q(\boldsymbol{\mathcal{A}}_P, \boldsymbol{r}_P)$ respectively. Appendix A provides a proof to the above statement for the general system of Multi-Input Multi-Output (MIMO) ports.

The moment matching property mentioned above is the main reason that Krylov subspace techniques are being classified as Implicit-Moment Matching, since matching the moments comes as a by-product of the projection operation and not as an explicit procedure that aims at calculating those moments as in (2.6).

If $\boldsymbol{\mathcal{P}}$ and $\boldsymbol{\mathcal{Q}}$ are chosen as above such that the reduced system matrix $\hat{\boldsymbol{\mathcal{A}}}$ is nonsingular, the technique is called two-sided process while if one matrix is chosen as above and the other is chosen arbitrarily such that the system matrix is nonsingular, the process is called one-sided process. One-sided process matches q moments of the output waveforms and provides better approximation of the system states trajectory. On the other hand, two-sided process has the advantage of matching $2q$ moments of the output waveform. Unlike the one-sided process, a two-sided process does not provide an accurate trajectory of the system states [27].

The following subsections present three techniques to compute the projection operator $\boldsymbol{\mathcal{P}}$ and $\boldsymbol{\mathcal{Q}}$. These techniques essentially rely on the Modified-Gram-Schmidt (MGS) orthogonalization process to compute those operators. However, given that the columns of the Krylov subspace are recursively related through matrix $\boldsymbol{\mathcal{A}}$, the MGS can be modified to take advantage of this recursive relation.

This modification ultimately results in two processes known as the *Arnoldi* in case of one-sided process or the *Lanczos* in case of two-sided process [27], [33]. It has been shown that both of these processes have superior numerical accuracy over the classical MGS since they do not require computing the columns of the Krylov subspace a priori.

[One-Sided] Arnoldi Process

In one-sided Arnoldi process, One projection operator is generated, namely matrix $\mathcal{Q}$, as defined in (2.16). Matrix $\mathcal{P}$ could be chosen arbitrarily, however it is not uncommon to choose matrix $\mathcal{P}$ the same as matrix $\mathcal{Q}$. Figure 2.5 depicts the Arnoldi algorithm which makes use of modified Gram-Schmidt (MGS) orthogonalizaion techniques. The projection operator $\mathcal{Q}$ generated in this process is orthonormal operator [34].

$$\mathcal{Q}^T\mathcal{Q} = \mathcal{I} \tag{2.19}$$

Algorithm 1: Computing $\mathcal{Q}$

input: $\mathcal{G}, \mathcal{C}, \boldsymbol{L}, \boldsymbol{B}, q$
output: $\mathcal{Q}$
begin
  Solve: $\mathcal{G}\tilde{q}_0 = \boldsymbol{B}$
  $q_0 \leftarrow \tilde{q}_0 / \|\tilde{q}_0\|$
  for $m \leftarrow 1$ **to** $q-1$ **do**
    Solve: $\mathcal{G}q_m = \mathcal{C}q_{m-1}$
    for $v = 0 \leftarrow$ **to** $m-1$ **do**
      $h \leftarrow q_v^T \tilde{q}_m$
      $\tilde{q}_m \leftarrow \tilde{q}_m - q_v h$
    $q_m \leftarrow \tilde{q}_m / \|\tilde{q}_m\|$
  return $\mathcal{Q}$
end

Figure 2.5: Pseudo-code description of the Arnoldi process

Lanczos Process

Lanczos process defines two projection operators as defined in (2.16) and (2.17). These two operators are biorthonormal operators, i.e.

$$\mathcal{P}^T\mathcal{Q} = \mathcal{I} \tag{2.20}$$

Figure 2.6 presents a modified algorithm to compute the projection operators [35], [36], [37]. This algorithm also makes use of the modified Gram-Schmidt process and has a better numerical performance when compared to the original one.

Algorithm 2: Computing $\mathcal{P}$, $\mathcal{Q}$

input: $\mathcal{G}, \mathcal{C}, \boldsymbol{L}, \boldsymbol{B}, q$
output: $\mathcal{P}$, $\mathcal{Q}$

begin
  $\mathcal{A} \leftarrow \mathcal{G}^{-1}\mathcal{C}$
  $\boldsymbol{p}_0 = \frac{sign(\boldsymbol{L}^T\boldsymbol{B})\boldsymbol{B}}{\sqrt{|\boldsymbol{L}^T\boldsymbol{B}|}}$
  $\boldsymbol{q}_0 = \frac{\boldsymbol{L}}{\sqrt{|\boldsymbol{L}^T\boldsymbol{B}|}}$
  for $m \leftarrow 1$ **to** $q-1$ **do**
    $\tilde{\boldsymbol{p}}_m \leftarrow \mathcal{A}\boldsymbol{p}_{m-1}$
    $\tilde{\boldsymbol{q}}_m \leftarrow \mathcal{A}^T\boldsymbol{q}_{m-1}$
    for $v = 0 \leftarrow$ **to** $m-1$ **do**
      $h1 \leftarrow \tilde{\boldsymbol{p}}_m^T\boldsymbol{q}_v$
      $\tilde{\boldsymbol{p}}_m \leftarrow \tilde{\boldsymbol{p}}_m - \boldsymbol{p}_v h1$
      $h2 \leftarrow \tilde{\boldsymbol{q}}_m^T\boldsymbol{p}_v$
      $\tilde{\boldsymbol{q}}_m \leftarrow \tilde{\boldsymbol{q}}_m - \boldsymbol{q}_v h2$
    $\boldsymbol{p}_m = \frac{sign(\tilde{\boldsymbol{p}}_m^T\tilde{\boldsymbol{q}}_m)\tilde{\boldsymbol{p}}_m}{\sqrt{\tilde{\boldsymbol{p}}_m^T\tilde{\boldsymbol{q}}_m}}$
    $\boldsymbol{q}_m = \frac{\tilde{\boldsymbol{q}}_m}{\sqrt{\tilde{\boldsymbol{p}}_m^T\tilde{\boldsymbol{q}}_m}}$
  return $\mathcal{P}$, $\mathcal{Q}$
end

Figure 2.6: Pseudo-code description of the Lanczos process

Two-Sided Arnoldi Process

This process although it defines two projection operators according to (2.16) and (2.17), each projection operator is not tangled with the other projection operator as in Lanczos process. Each operator is an orthonormal operator and is a direct result of applying one-sided Arnoldi (shown in Figure 2.5) process. The authors in [27] show that two-sided Arnoldi process has a better performance over both Lanczos and one-sided Arnoldi processes.

2.2.3 Passivity Consideration

Passivity is an important property that needs to be preserved in the resulting reduced order model. The reason for that, non-passive systems can become unstable under different loading conditions. This instability can introduce artificial numerical oscillation during transient simulation.

In the *PRIMA* algorithm [13] , [38] it was proved that one-sided process, in which $\mathcal{P} = \mathcal{Q}^T$ can guarantee the passivity of the reduced system. Due to this desirable feature, the Arnoldi process has been adopted in this thesis as the main tool to reduce the linear circuits.

Chapter 3

Sensitivity Analysis

In this chapter, we present background information on different approaches to computing sensitivity information. The term "sensitivity analysis" usually refers to computing the derivative of one or more network responses w.r.t. one or more network design parameters. Derivation of sensitivity information could be categorized in two main categories; Adjoint Sensitivity analysis and Direct Differentiation analysis. This chapter describes the concepts behind these two approaches and illustrates their applications in computing both time- and frequency-domain sensitivities.

3.1 Adjoint Sensitivity Analysis

In adjoint sensitivity analysis, sensitivity information is obtained by considering the simulation of two networks:

- The first network is the original network. Response of this network either in the frequency domain or time domain is obtained as a part of the initial simulation phase.
- The second network, known as the adjoint network, is derived from the original network

through some well-known techniques e.g. [5].

Both of these networks are simulated for specific frequency/time range and sensitivity information is obtained based on the results of these simulations.

Adjoint sensitivity is favored in CAD framework for its inherent ability of computing the sensitivity information using only the above two system solutions regardless of the number of the design parameters. However, this information is computed for only one or few targeted network waveforms.

3.1.1 Frequency-Domain Adjoint Sensitivity

Adjoint sensitivity in frequency domain tackles sensitivity information of circuits having linear elements only. As illustrated in chapter 2, any linear circuit can be described in frequency domain using the MNA formulation as follows:

$$(\boldsymbol{\mathcal{G}}_{\pi_i}(\lambda_{i,j}) + s\boldsymbol{\mathcal{C}}_{\pi_i}(\lambda_{i,j}))\boldsymbol{X}_{\pi_i}(s, \lambda_{i,j}) = \boldsymbol{\mathcal{B}}\boldsymbol{U}_{\pi_i}(s) \tag{3.1}$$

where the subscript π_i, (with i assumed being an index $i = 1, \ldots N$) has been appended to each term in (3.1) to emphasize that the linear circuit can be embedded as a linear subnetwork in a larger network, which contain similar/dissimilar $N-1$ linear circuits. This notation will be beneficial in later chapters where the main contribution of the thesis will be presented. $\lambda_{i,j}$ in (3.1) refers to the j^{th} parameter, for which the sensitivity information is to be computed, of network π_i, and s is the Laplace variable.

In order to compute the sensitivity information for a variable V of subnetwork π_i, we opt to write this variable V as follows:

$$V = \boldsymbol{d}^T\boldsymbol{X}_{\pi_i}(s, \lambda_{i,j}) \tag{3.2}$$

where $\boldsymbol{d}^T$ is a selector vector that has unity in the location corresponding to network variable V and zeros otherwise. The sensitivity of V w.r.t. $\lambda_{i,j}$ could be obtained by differentiating (3.2) as follows:

$$\frac{\partial V}{\partial \lambda_{i,j}} = -\underbrace{\boldsymbol{d}^T \boldsymbol{\mathcal{A}}^{-1}}_{\boldsymbol{X}_a^T} \frac{\partial \boldsymbol{\mathcal{A}}}{\partial \lambda_{i,j}} \boldsymbol{X}(s, \lambda_{i,j}) \tag{3.3}$$

where $\boldsymbol{\mathcal{A}} = \boldsymbol{\mathcal{G}}_{\pi_i}(\lambda_{i,j}) + s\boldsymbol{\mathcal{C}}_{\pi_i}(\lambda_{i,j})$.

Computation of the Equation (3.3) could be divided into three terms; two of which are independent of any differentiation w.r.t. the sensitivity parameter $\lambda_{i,j}$. The first term is the term on the right of the right-hand side of (3.3). This term represents the result of the frequency-domain analysis performed on the original network at frequency ω, $s = 2\pi j\omega$, and its computation involves one L/U decomposition of the matrix $\boldsymbol{\mathcal{A}}$ and one F/B substitution with the vector $\boldsymbol{\mathcal{B}}\boldsymbol{U}_\pi(s)$. The other term is $\boldsymbol{X}_a$ and is usually known as the adjoint variable. Such a term can be obtained through solving thee following system:

$$\boldsymbol{\mathcal{A}}^T \boldsymbol{X}_a(s, \lambda_{i,j}) = \boldsymbol{d} \tag{3.4}$$

To solve for $\boldsymbol{X}_a$, one usually needs to perform L/U decomposition of the matrix $\boldsymbol{\mathcal{A}}^T$ followed by a F/B substitution with vector $\boldsymbol{d}$. However, given that the L/U factors of the matrix $\boldsymbol{\mathcal{A}}$ are already computed beforehand during the frequency-domain analysis phase, and since $\boldsymbol{\mathcal{A}}^T = (\boldsymbol{\mathcal{L}}\boldsymbol{\mathcal{U}})^T = \boldsymbol{\mathcal{U}}^T\boldsymbol{\mathcal{L}}^T$, then X_a can be obtained through an additional F/B substitution using the matrices $\boldsymbol{\mathcal{U}}^T$ and $\boldsymbol{\mathcal{L}}^T$ and the vector $\boldsymbol{d}$. Thus, the total cost incurred to compute the sensitivity information w.r.t. one variable is one L/U factorization, two F/B substitution, matrix-vector multiplication, and vector-vector multiplication. The dominant factor would be of order $\mathrm{O}(N^2)$ which results from the additional F/B substitution. However, for subsequent parameters, the incurred computations per parameter boils down to one F/B substitution, matrix-vector multiplication, and vector-vector multiplication. The latter cost offers greater savings in terms of

computation time, however, the order of the computation has not changed.

3.1.2 Time-Domain Sensitivity

The concept of adjoint sensitivity analysis is also applicable to time domain sensitivity computation. Here, the objective is to compute the sensitivity (derivatives) of a network variable(s) (or an objective or cost function defined in terms of these network variables) over a period of time that extends from $t = 0$ to, say $t = t^*$. To simplify the presentation, assume that our goal is to compute such a derivative for a single network variable, in a nonlinear network, θ_i, described by the following equations:

$$\boldsymbol{C}_{\theta_i}\dot{\boldsymbol{x}}_{\theta_i}(t) + \boldsymbol{G}_{\theta_i}\boldsymbol{x}_{\theta_i}(t) + \boldsymbol{f}_{\theta_i}(\boldsymbol{x}_{\theta_i}(t)) = \boldsymbol{B}_{\theta_i}\boldsymbol{u}_{\theta_i}(t) \tag{3.5}$$

Let $\lambda_{i,j}$ stand for the design parameter in θ_i, with $j = 1, \ldots, k$ and denote the network variable whose sensitivity (derivatives) w.r.t. all $\lambda_{i,j}$ is described by $\boldsymbol{x}_{\theta_i,l}(t)$ and assume that it appears in the l^{th} component in $\boldsymbol{x}_{\theta_i}(t)$, the time-dependent vector for all network variables in θ_i. Hence, $\boldsymbol{x}_{\theta_i,l}(t)$ can be written as:

$$\boldsymbol{x}_{\theta_i,l}(t) = \boldsymbol{d}^T \int_0^t \dot{\boldsymbol{x}}_{\theta_i}(z)dz \tag{3.6}$$

The above integral is evaluated such that $\boldsymbol{x}_{\theta_i}(t)$, $\dot{\boldsymbol{x}}_{\theta_i}(t)$ satisfy the network constitutive equations given in (3.5). This leads to the following Lagrangian function L:

$$L(t) = \boldsymbol{d}^T\dot{\boldsymbol{x}}_{\theta_i}(t) + \boldsymbol{\mu}^T(t)\left(\boldsymbol{C}_{\theta_i}\dot{\boldsymbol{x}}_{\theta_i}(t) + \boldsymbol{G}_{\theta_i}\boldsymbol{x}_{\theta_i}(t) + \boldsymbol{f}_{\theta_i}(\boldsymbol{x}_{\theta_i}(t)) - \boldsymbol{B}_{\theta_i}\boldsymbol{u}_{\theta_i}(t)\right) \tag{3.7}$$

where $\boldsymbol{\mu}(t)$ is a vector of Lagrange multipliers and superscript T denotes the transpose opera-

tion. Hence, $\partial \boldsymbol{x}_{\theta_i,l}(t)/\partial \lambda_{i,j}$ is given by:

$$\frac{\partial \boldsymbol{x}_{\theta_i,l}(t)}{\partial \lambda_{i,j}} = \frac{\partial}{\partial \lambda_{i,j}} \int_0^t L(z)dz \tag{3.8}$$

Differentiating (3.8) and rearranging the integral would result in[1]:

$$\begin{aligned} \frac{\partial \boldsymbol{x}(t)}{\partial \lambda} = \int_0^t & \left\{ \left(\boldsymbol{d}^T + \boldsymbol{\mu}^T \boldsymbol{\mathcal{C}}\right) \frac{\partial \dot{\boldsymbol{x}}}{\partial \lambda} \right. \\ & \left. + \boldsymbol{\mu}^T \left(\frac{\partial \boldsymbol{\mathcal{C}}}{\partial \lambda}\dot{\boldsymbol{x}} + \frac{\partial \boldsymbol{\mathcal{G}}}{\partial \lambda}\boldsymbol{x} + \boldsymbol{\mathcal{G}}\frac{\partial \boldsymbol{x}}{\partial \lambda} + \frac{\partial \boldsymbol{f}(\boldsymbol{x})}{\partial \lambda} + \frac{\partial \boldsymbol{f}(\boldsymbol{x})}{\partial \boldsymbol{x}}\frac{\partial \boldsymbol{x}}{\partial \lambda} \right) \right\} dz \end{aligned} \tag{3.9}$$

Now we define a boundary point in time at $t = t^*$ where the integral defined in Equation (3.9) is evaluated. The evaluation could be computed by integrating the first part which is dependent on $\partial \dot{\boldsymbol{x}}(t)/\partial \lambda$ yields the following set of equations:

$$\begin{aligned} \left.\frac{\partial \boldsymbol{x}(t)}{\partial \lambda}\right|_{t=t^*} = & \left(\left(\boldsymbol{d}^T + \boldsymbol{\mu}^T \boldsymbol{\mathcal{C}}\right) \frac{\partial \boldsymbol{x}}{\partial \lambda} \right)\Bigg|_0^{t^*} + \int_0^{t^*} \boldsymbol{\mu}^T \left(\frac{\partial \boldsymbol{\mathcal{C}}}{\partial \lambda}\dot{\boldsymbol{x}} + \frac{\partial \boldsymbol{\mathcal{G}}}{\partial \lambda}\boldsymbol{x} + \frac{\partial \boldsymbol{f}(\boldsymbol{x})}{\partial \lambda} \right) dz \\ & + \int_0^{t^*} \underbrace{\left\{ \boldsymbol{\mu}^T \left(\boldsymbol{\mathcal{G}} + \frac{\partial \boldsymbol{f}(\boldsymbol{x})}{\partial \boldsymbol{x}} \right) - \frac{d\boldsymbol{\mu}^T}{dz}\boldsymbol{\mathcal{C}} \right\}}_{\zeta} \frac{\partial \boldsymbol{x}}{\partial \lambda} dz \end{aligned} \tag{3.10}$$

The choice of the Lagrange multiplier $\boldsymbol{\mu}$ can be made to set the term ζ in (3.10) to zero, leading to the following system:

$$\boldsymbol{\mu}^T \left(\boldsymbol{\mathcal{G}} + \frac{\partial \boldsymbol{f}(\boldsymbol{x})}{\partial \boldsymbol{x}} \right) - \frac{d\boldsymbol{\mu}^T}{dt}\boldsymbol{\mathcal{C}} = \boldsymbol{0} \tag{3.11}$$

[1]Subscripts θ_i, l, i, j and the suffixes (t) and (z) will be dropped, when they are clearly understood, for the ease of presentation

subject to the terminal condition

$$\boldsymbol{d}^T + \boldsymbol{\mu}^T(t^*)\boldsymbol{C} = \mathbf{0} \tag{3.12}$$

Thus Equation (3.9) reduces to:

$$\begin{aligned}\frac{\partial \boldsymbol{x}}{\partial \lambda} &= -\left(\boldsymbol{d}^T + \boldsymbol{\mu}^T(0)\boldsymbol{C}\right)\frac{\partial \boldsymbol{x}(0)}{\partial \lambda} + \\ &\int_0^{t^*} \boldsymbol{\mu}^T\left(\frac{\partial \boldsymbol{C}}{\partial \lambda}\dot{\boldsymbol{x}} + \frac{\partial \boldsymbol{G}}{\partial \lambda}\boldsymbol{x} + \frac{\partial \boldsymbol{f}(\boldsymbol{x})}{\partial \lambda} - \frac{\partial \boldsymbol{b}}{\partial \lambda}\right)dt\end{aligned} \tag{3.13}$$

Hence, the derivative $\partial \boldsymbol{x}/\partial \lambda$, the desired sensitivity, can be obtained by first integrating (3.5) forward in time to obtain $\boldsymbol{x}(t)$, and then integrating (3.11) backward in time subject to (3.12) to obtain $\boldsymbol{\mu}(t)$ and then using both results to obtain the integral in (3.13).

3.2 Direct Differentiation Sensitivity Analysis

Sensitivity analysis based on direct differentiation is preferred over adjoint analysis in situation where the circuit designer is interested in the sensitivity of all network variables w.r.t. few design parameters. However, this is typically an unlikely situation, since the goal is usually to compute the sensitivity of few variables that represent important metrics, such as the voltage at a receiver end of a transmission line cable, w.r.t. all design parameters.

As its name implies, direct differentiation operated by taking the derivative of the network

equations in (3.5) w.r.t. a given design parameter λ to obtain:

$$\boldsymbol{C}\frac{\partial \dot{\boldsymbol{x}}(t)}{\partial \lambda} + \left\{\boldsymbol{G} + \frac{\partial \boldsymbol{f}(\boldsymbol{x}(t))}{\partial \boldsymbol{x}(t)}\right\}\frac{\partial \boldsymbol{x}(t)}{\partial \lambda} = - \left\{\frac{\partial \boldsymbol{C}}{\partial \lambda}\dot{\boldsymbol{x}}(t) + \frac{\partial \boldsymbol{G}}{\partial \lambda}\boldsymbol{x}(t) + \frac{\partial \boldsymbol{f}(\boldsymbol{x}(t))}{\partial \lambda}\right\} + \underbrace{\boldsymbol{B}\frac{\partial \boldsymbol{u}(t)}{\partial \lambda}}_{=\mathbf{0}} \tag{3.14}$$

The solution obtained by integrating the above system gives the desired sensitivity. It is obvious here that in order to compute the sensitivity w.r.t. say k design parameters, one need to perform k system analyses similar to (3.14) but with different stimulus vectors on the right hand side.

It is possible to achieve some computational savings by storing the Jacobian matrix information, such as its L/U factors, at the different time points taken during the integration and use them in the analysis of (3.14). However, this in addition to increasing the memory storage demands of the techniques, implicitly assume that the numerical solver used to solve (3.14) will step over exactly the same time points used or encountered during the integration, which is usually highly unlikely since the different stimulus vectors necessitates different time steps to control the local truncation error appropriately [5].

3.3 Discussion

This chapter presented a brief review of the two main approaches that can be used to perform sensitivity analysis in both frequency domain and time domain, namely adjoint-based and direct differentiation, and illustrated the advantages/disadvantages in both approaches. The focus of this thesis, however, is on employing the adjoint-based approach to compute the time-domain sensitivity of circuits in which one or more linear subnetworks are being described by a projection-based MOR algorithm such as ones described in Section 2.2.2.

Equations (3.13) and (3.14) clearly show that, whether an adjoint-based or a direct differentiation technique is being adopted as the sensitivity computation approach, both approaches require prior knowledge of the derivative of the system matrices $\boldsymbol{\mathcal{G}}$ and $\boldsymbol{\mathcal{C}}$ w.r.t. the design parameter λ. As the next chapter will demonstrate, this requirement poses a peculiar difficulty in the situation where MOR has been applied on a linear subnetwork within the circuit, if the design parameter λ is a parameter within this subnetwork.

Chapter 4

Sensitivity Analysis Using Model-Order Reduction

This chapter examines the application of the concept of Model-Order Reduction (MOR) to adjoint-based sensitivity analysis. Both time- and frequency-domain adjoint sensitivity using MOR will be discussed. A brief review of the earlier techniques that aimed at combining adjoint sensitivity analysis with MOR will be described. Finally, an emphasis will be placed on the main issue that hinders the application of the adjoint principal in MOR to time-domain sensitivity based on the adjoint method.

4.1 Structure of Networks with Reduced-Order Models

To emphasize the main issues involved in combining MOR with adjoint-based sensitivity analysis, this section considers first a general network which includes some large subnetworks that have been reduced using one of the MOR techniques presented in Chapter 2, such as the PRIMA algorithm. Attempting to apply sensitivity analysis on such a system will highlight the main difficulties encountered therein and will also provide the necessary background on

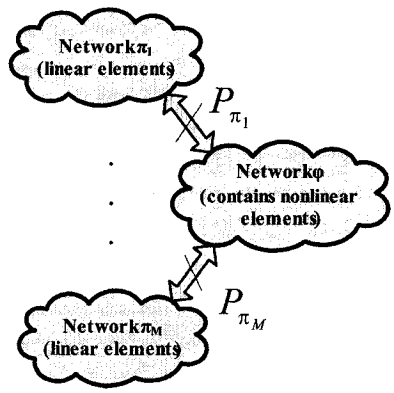


Figure 4.1: Electrical network θ

existing approaches that have been introduced to tackle this problem.

For that purpose, consider a network θ which is composed of M linear subnetworks (denoted here by π_i where $i = 1, \ldots, M$) and let ϕ be the subnetwork that contains all lumped linear and nonlinear elements. Assume that each subnetwork π_i is characterized by large order, which typically arises from applying some finite discretization techniques on a microwave device or a distributed element, such as a transmission line. Subnetwork ϕ contains the nonlinear lumped components that could represent inverters, terminating buffers, or driver circuitry. Figure (4.1) describes a schematic representation for the network θ, which shows that each subnetwork π_i is interconnected through P_{π_i} ports with the rest of the network. Without loss of generality, each subnetwork π_i is assumed to be represented by a large network of lumped **RLC** components. Hence the MNA formulation method can be used to describe subnetwork π_i by the following system of differential equations:

$$\boldsymbol{\mathcal{G}}_{\pi_i}\boldsymbol{x}_{\pi_i}(t) + \boldsymbol{\mathcal{C}}_{\pi_i}\frac{d\boldsymbol{x}_{\pi_i}(t)}{dt} = \boldsymbol{\mathcal{B}}_{\pi_i}\boldsymbol{u}(t) \tag{4.1}$$

in the time domain, or by the following system of algebraic equations :

$$\boldsymbol{\mathcal{G}}_{\pi_i}\boldsymbol{X}_{\pi_i}(s) + s\boldsymbol{\mathcal{C}}_{\pi_i}\boldsymbol{X}_{\pi_i}(s) = \boldsymbol{\mathcal{B}}_{\pi_i}\boldsymbol{U}(s) \tag{4.2}$$

in the frequency domain. Description of the various terms appearing in (4.1) or (4.2) have been presented earlier in Section 2.1.1. The currents at the ports of subnetwork π_i can be obtained using $\boldsymbol{B}_{\pi_i}$ as follows:

$$\boldsymbol{i}_{\pi_i}(t) = \boldsymbol{\mathcal{B}}_{\pi_i}^T \boldsymbol{x}_{\pi_i}(t) \tag{4.3}$$

Furthermore, assume that MOR using a matrix projection algorithm has been applied to each subnetwork π_i, and let $\hat{\boldsymbol{\mathcal{G}}}_{\pi_i}$, $\hat{\boldsymbol{\mathcal{C}}}_{\pi_i} \in \mathbb{R}^{q_{\pi_i} \times q_{\pi_i}}$, and $\hat{\boldsymbol{B}}_{\pi_i} \in \mathbb{R}^{q_{\pi_i} \times P_{\pi_i}}$ stand for the resulting reduced matrices, i.e.:

$$\begin{aligned} \hat{\boldsymbol{\mathcal{C}}}_{\pi_i} &= \boldsymbol{\mathcal{Q}}_{\pi_i}^T \boldsymbol{\mathcal{C}}_{\pi_i} \boldsymbol{\mathcal{Q}}_{\pi_i} \\ \hat{\boldsymbol{\mathcal{G}}}_{\pi_i} &= \boldsymbol{\mathcal{Q}}_{\pi_i}^T \boldsymbol{\mathcal{G}}_{\Phi} \boldsymbol{\mathcal{Q}}_{\pi_i} \\ \hat{\boldsymbol{\mathcal{B}}}_{\pi_i} &= \boldsymbol{\mathcal{Q}}_{\pi_i}^T \boldsymbol{\mathcal{B}}_{\pi_i} \end{aligned} \tag{4.4}$$

where $\boldsymbol{\mathcal{Q}}_{\pi_i}$ is an orthogonal basis that spans the Krylov subspace $\mathrm{K}_r(\boldsymbol{\mathcal{G}}_{\pi_i}^{-1}\boldsymbol{\mathcal{C}}_{\pi_i}, \boldsymbol{\mathcal{G}}_{\pi_i}^{-1}\boldsymbol{\mathcal{B}}_{\pi_i})$ with $r = \left\lceil \frac{N_{\pi_i}}{q_{\pi_i}} \right\rceil$ and N_{π_i} (q_{π_i}) is the size of the original (reduced) system.

The reduced system in (4.4) can be employed to provide a representation of π_i in the reduced domain. Such a representation will take the following form:

$$\begin{aligned} \hat{\boldsymbol{\mathcal{G}}}_{\pi_i} \hat{\boldsymbol{x}}_{\pi_i}(t) + \hat{\boldsymbol{\mathcal{C}}}_{\pi_i} \frac{d\hat{\boldsymbol{x}}_{\pi_i}(t)}{dt} &= \hat{\boldsymbol{\mathcal{B}}}_{\pi_i} \boldsymbol{u}(t) \\ \boldsymbol{i}_{\pi_i}(t) &= \hat{\boldsymbol{\mathcal{B}}}_{\pi_i}^T \hat{\boldsymbol{x}}_{\pi_i}(t) \end{aligned} \tag{4.5}$$

in the time domain, or the following algebraic formulation:

$$\begin{aligned} \hat{\boldsymbol{\mathcal{G}}}_{\pi_i} \hat{\boldsymbol{X}}_{\pi_i}(s) + \hat{\boldsymbol{\mathcal{C}}}_{\pi_i} \hat{\boldsymbol{X}}_{\pi_i}(s) &= \hat{\boldsymbol{\mathcal{B}}}_{\pi_i} \boldsymbol{U}(s) \\ \boldsymbol{I}_{\pi_i}(s) &= \hat{\boldsymbol{\mathcal{B}}}_{\pi_i}^T \hat{\boldsymbol{X}}_{\pi_i}(s) \end{aligned} \tag{4.6}$$

in the frequency domain. Given that each subnetwork π_i has undergone an MOR process that

resulted in a system similar to the on shown in (4.5) or (4.6), the goal now shifts to finding a global formulation for the entire network θ. The presence of lumped nonlinear elements, however, mandates that the time domain is the natural domain for a global formulation of θ which can be represented in the following form:

$$\tilde{\boldsymbol{\mathcal{G}}}\tilde{\boldsymbol{x}}(t) + \tilde{\boldsymbol{\mathcal{C}}}\frac{d\tilde{\boldsymbol{x}}}{dt} + \tilde{\boldsymbol{f}}(\tilde{\boldsymbol{x}}(t)) = \tilde{\boldsymbol{\mathcal{B}}}\tilde{\boldsymbol{u}}(t) \tag{4.7}$$

where the matrices $\tilde{\boldsymbol{\mathcal{G}}}$ and $\tilde{\boldsymbol{\mathcal{C}}}$ contain the reduced system matrices $\hat{\boldsymbol{\mathcal{G}}}_{\pi_i}$, $\hat{\boldsymbol{\mathcal{C}}}_{\pi_i}$, and $\hat{\boldsymbol{\mathcal{B}}}_{\pi_i}$ for each subnetwork π_i, in addition to the contribution of the linear elements of the subnetwork ϕ. A general form of the structure of those matrices can be described as shown in (4.8):

$$\begin{aligned}
\tilde{\boldsymbol{\mathcal{G}}} &= \begin{bmatrix} \boldsymbol{\mathcal{G}}_\phi & \boldsymbol{\mathcal{D}}_{\pi_1}\hat{\boldsymbol{\mathcal{B}}}_{\pi_1}^T & \cdots & \boldsymbol{\mathcal{D}}_{\pi_M}\hat{\boldsymbol{\mathcal{B}}}_{\pi_M}^T \\ -\hat{\boldsymbol{\mathcal{B}}}_{\pi_1}\boldsymbol{\mathcal{D}}_{\pi_1}^T & \hat{\boldsymbol{\mathcal{G}}}_{\pi_1} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ -\hat{\boldsymbol{\mathcal{B}}}_{\pi_M}\boldsymbol{\mathcal{D}}_{\pi_M}^T & \mathbf{0} & \cdots & \hat{\boldsymbol{\mathcal{G}}}_{\pi_M} \end{bmatrix}, \\
\tilde{\boldsymbol{\mathcal{C}}} &= \begin{bmatrix} \boldsymbol{\mathcal{C}}_\phi & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \hat{\boldsymbol{\mathcal{C}}}_{\pi_1} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \hat{\boldsymbol{\mathcal{C}}}_{\pi_M} \end{bmatrix}
\end{aligned} \tag{4.8}$$

where $\boldsymbol{\mathcal{D}}_{\pi_i} \in \mathbb{R}^{N_\phi \times N_{\pi_i}}$ are selector matrices with elements in $\{0, 1\}$ that map the currents at the ports of each subnetwork π_i to the space of network variables in θ.

The variables $\tilde{\boldsymbol{x}}(t)$ contain the variables from the MNA formulation of the network ϕ, in

addition to all other variables used in representing the reduced-order systems, i.e.:

$$\tilde{\boldsymbol{x}}(t) = \begin{bmatrix} \boldsymbol{x}_\phi(t) \\ \hat{\boldsymbol{x}}_{\pi_1}(t) \\ \vdots \\ \hat{\boldsymbol{x}}_{\pi_M}(t) \end{bmatrix} \tag{4.9}$$

where $\boldsymbol{x}_\phi(t)$ collects all MNA variables arising from the subnetwork ϕ and $\hat{\boldsymbol{x}}_{\pi_i}(t)$ is the internal variables of subnetwork π_i when projected in the reduced domain.

Let $\tilde{\boldsymbol{f}}(.)$ be a vector whose entries are the nonlinear functions of the nonlinear elements in ϕ and defined as follows:

$$\tilde{\boldsymbol{f}}(\tilde{\boldsymbol{x}}(t)) = \begin{bmatrix} \boldsymbol{f}_\phi(\boldsymbol{x}_\phi(t)) \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix} \tag{4.10}$$

where the zero entries result from the absence of nonlinear elements in all the linear subnetworks π_i.

Matrix $\tilde{\boldsymbol{\mathcal{B}}}$ is constructed as follows:

$$\tilde{\boldsymbol{\mathcal{B}}} = \begin{bmatrix} \boldsymbol{\mathcal{B}}_\phi & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix} \tag{4.11}$$

where $\boldsymbol{\mathcal{B}}_\phi$ is the matrix that represents source-to-port maps in θ. These sources are assumed to be contained within the subnetwork ϕ.

Finally the vector $\tilde{\boldsymbol{u}}$ is obtained in similar manner, i.e.:

$$\tilde{\boldsymbol{u}}(t) = \begin{bmatrix} \boldsymbol{u}_\phi(t) \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix} \tag{4.12}$$

4.2 Sensitivity Analysis for Reduced Systems

Having described the general form expected for the formulation of a network, in which several subnetworks have been represented by reduced models, sets the stage for considering sensitivity analysis. It should be noted that the task of sensitivity analysis has traditionally been considered for classical circuit networks which do not include the networks described by reduced models obtained from projection techniques.

We use $\lambda_{i,j}$ to denote a parameter with respect to which sensitivity is required. The subscript i indicates that $\lambda_{i,j}$ belongs to subnetwork π_i and subscript j, where $j = 1, \ldots, k_{\pi_i}$, means the j^{th} design parameter within π_i is considered. In typical situations, one is usually interested in the sensitivity of a specific variable such as node voltage, in the network θ or at most few sets of variables. Without loss of generality it will be assumed that sensitivity of a specific variable, denoted here by V, is being sought. V can be expressed using the notion of the selector vector, $\boldsymbol{d}$, where $\boldsymbol{d}$ is a vector with only one unity entry and zero otherwise. Using $\boldsymbol{d}$, V can be expressed as:

$$V = \boldsymbol{d}^T \tilde{\boldsymbol{x}}(t) = \begin{bmatrix} \boldsymbol{e}_m^T & \mathbf{0} & \ldots & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_\phi(t) \\ \hat{\boldsymbol{x}}_{\pi_1}(t) \\ \vdots \\ \hat{\boldsymbol{x}}_{\pi_M}(t) \end{bmatrix} \tag{4.13}$$

where $\boldsymbol{e}_m$ is the m^{th} column in the identity matrix $\boldsymbol{\mathcal{I}}_{N_\phi \times N_\phi}$ and $1 \leqslant m \leqslant N_\phi$.

Notice that according to the above formulation, $\boldsymbol{d}$ is independent of the sensitivity parameter $\lambda_{i,j}$. Also this formulation implicitly assumes that we are only interested in the variables within network ϕ, while the internal variables within subnetworks π_i are of no interest. In fact, this is typically the case, since these internal variables only emerge as a result of applying some discretization technique on a distributed device and therefore have no direct physical interpretation.

On the other hand, it is usually the variables within $\boldsymbol{x}_\phi(t)$ that represent target waveforms such as the voltage at a buffering circuit. It should be noted that, had it been the case that sensitivity of internal variable is the goal, then it will be necessary to re-map the reduced variables of subnetwork π_i to their original domain using the projection operator $\boldsymbol{\mathcal{Q}}_{\pi_i}$. This would have entailed modifying the selector vector $\boldsymbol{d}$ to be similar to $\begin{bmatrix} \mathbf{0} & \mathbf{0} & (\boldsymbol{e}_m^T \boldsymbol{\mathcal{Q}}_{\pi_i}) & \mathbf{0} & \dots & \mathbf{0} \end{bmatrix}$ and consequently made it dependent on $\lambda_{i,j}$ since $\boldsymbol{\mathcal{Q}}_{\pi_i}$ is dependent on $\lambda_{i,j}$ as will be shown later.

Now proceeding from the assumption that $\boldsymbol{d}$ is independent of $\lambda_{i,j}$, the sensitivity of V w.r.t. $\lambda_{i,j}$ can be obtained from:

$$\frac{\partial V}{\partial \lambda_{i,j}} = \begin{bmatrix} \boldsymbol{e}_m^T & \mathbf{0} & \dots & \mathbf{0} \end{bmatrix} \frac{\partial}{\partial \lambda_{i,j}} \begin{bmatrix} \boldsymbol{x}_\phi(t) \\ \hat{\boldsymbol{x}}_{\pi_1}(t) \\ \vdots \\ \hat{\boldsymbol{x}}_{\pi_M}(t) \end{bmatrix} \tag{4.14}$$

The objective now is to consider the idea of adjoint sensitivity analysis in computing $\partial V / \partial \lambda_{i,j}$ in both frequency domain and time domain.

4.2.1 Frequency-Domain Adjoint Sensitivity Analysis Using MOR

To obtain frequency-domain sensitivity using the adjoint-based technique for the variable V directly from the frequency domain requires formulating the network θ in the frequency domain. This however, requires removing the assumption that θ has nonlinear lumped components in order to enable such a formulation. Otherwise, frequency-domain sensitivity will have to be computed indirectly through first running a time-domain adjoint sensitivity analysis and then using a discrete or fast Fourier transform to obtain the frequency-domain data. The issues involved in the time-domain sensitivity analysis are similar to those raised by the frequency domain and will discussed in the next subsection. The absence of nonlinear elements in θ enables writing it in the frequency domain as follows:

$$\tilde{\boldsymbol{\mathcal{G}}}\tilde{\boldsymbol{X}}(s) + s\tilde{\boldsymbol{\mathcal{C}}}\tilde{\boldsymbol{X}}(s) = \tilde{\boldsymbol{\mathcal{B}}}\boldsymbol{U}(s) \tag{4.15}$$

As discussed earlier, the variable of interest V is given in the frequency domain by:

$$V = \boldsymbol{d}^T\tilde{\boldsymbol{X}}(s, \lambda_{i,j}) \tag{4.16}$$

Now we proceed to find the derivative of V w.r.t. $\lambda_{i,j}$. The differentiation of (4.16) leads to the following equation:

$$\frac{\partial V}{\partial \lambda_{i,j}} = \boldsymbol{d}^T\frac{\partial\tilde{\boldsymbol{X}}(s, \lambda_{i,j})}{\partial \lambda_{i,j}} \tag{4.17}$$

Differentiating (4.15) and substituting for $\frac{\partial\tilde{\boldsymbol{X}}(\lambda_{i,j})}{\partial\lambda_{i,j}}$ in (4.16) yields the following:

$$\frac{\partial V}{\partial \lambda_{i,j}} = -\underbrace{\tilde{\boldsymbol{d}}^T\tilde{\boldsymbol{\mathcal{A}}}^{-1}}_{\tilde{\boldsymbol{X}}_a^T}\frac{\partial\tilde{\boldsymbol{\mathcal{A}}}}{\partial\lambda_{i,j}}\tilde{\boldsymbol{X}}(s, \lambda_{i,j}) \tag{4.18}$$

where $\tilde{\boldsymbol{\mathcal{A}}} = \tilde{\boldsymbol{\mathcal{G}}}_{\pi_i}(\lambda_{i,j}) + s\tilde{\boldsymbol{\mathcal{C}}}_{\pi_i}(\lambda_{i,j})$. The term $\boldsymbol{d}^T\tilde{\boldsymbol{\mathcal{A}}}^{-1}$ defines the adjoint variables $\tilde{\boldsymbol{X}}_a$ which

is the solution of the adjoint system:

$$(\tilde{\boldsymbol{\mathcal{G}}}^T + s\tilde{\boldsymbol{\mathcal{C}}}^T)\tilde{\boldsymbol{X}}_a = \boldsymbol{d} \tag{4.19}$$

Thus to apply the principle of adjoint sensitivity analysis in computing $\partial V/\partial\lambda_{i,j}$, one will be faced with computing $\partial\tilde{\boldsymbol{\mathcal{A}}}/\partial\lambda_{i,j}$ which necessitates computing the derivative of $\tilde{\boldsymbol{\mathcal{G}}}_{\pi_i}$, $\tilde{\boldsymbol{\mathcal{C}}}_{\pi_i}$, and $\tilde{\boldsymbol{\mathcal{B}}}_{\pi_i}$ w.r.t. $\lambda_{i,j}$. These matrices, however, are constructed using the projection operator $\boldsymbol{\mathcal{Q}}_{\pi_i}$ which is, in turn, dependent on $\lambda_{i,j}$ but rather in a complicated manner. This is a consequence of having to obtain $\boldsymbol{\mathcal{Q}}_{\pi_i}$ from an iterative process as shown in Chapter 2.

In other words, it will be necessary to compute $\partial\tilde{\boldsymbol{\mathcal{G}}}/\partial\lambda_{i,j}$, $\partial\tilde{\boldsymbol{\mathcal{C}}}/\partial\lambda_{i,j}$, and $\partial\tilde{\boldsymbol{\mathcal{B}}}/\partial\lambda_{i,j}$. Using the chain rule of differentiation, these derivatives can be written as:

$$\begin{aligned}
\frac{\partial\hat{\boldsymbol{\mathcal{G}}}}{\partial\lambda_{i,j}} &= \frac{\partial\boldsymbol{\mathcal{Q}}^T}{\partial\lambda_{i,j}}\boldsymbol{\mathcal{G}}\boldsymbol{\mathcal{Q}} + \boldsymbol{\mathcal{Q}}^T\frac{\partial\boldsymbol{\mathcal{G}}}{\partial\lambda_{i,j}}\boldsymbol{\mathcal{Q}} + \boldsymbol{\mathcal{Q}}^T\boldsymbol{\mathcal{G}}\frac{\partial\boldsymbol{\mathcal{Q}}}{\partial\lambda_{i,j}} \\
\frac{\partial\hat{\boldsymbol{\mathcal{C}}}}{\partial\lambda_{i,j}} &= \frac{\partial\boldsymbol{\mathcal{Q}}^T}{\partial\lambda_{i,j}}\boldsymbol{\mathcal{C}}\boldsymbol{\mathcal{Q}} + \boldsymbol{\mathcal{Q}}^T\frac{\partial\boldsymbol{\mathcal{C}}}{\partial\lambda_{i,j}}\boldsymbol{\mathcal{Q}} + \boldsymbol{\mathcal{Q}}^T\boldsymbol{\mathcal{C}}\frac{\partial\boldsymbol{\mathcal{Q}}}{\partial\lambda_{i,j}} \\
\frac{\partial\hat{\boldsymbol{\mathcal{B}}}}{\partial\lambda_{i,j}} &= \frac{\partial\boldsymbol{\mathcal{Q}}^T}{\partial\lambda_{i,j}}\boldsymbol{\mathcal{B}}
\end{aligned} \tag{4.20}$$

which shows clearly the need to compute the derivative of the projection operator $\boldsymbol{\mathcal{Q}}_{\pi_i}$. However, as noted by the authors in [17], a direct computation of this term is "a difficult and cumbersome task." To avoid having to compute such a derivative the authors in [17] describe an approach that relies on finding an auxiliary basis $\boldsymbol{\mathcal{Q}}_a$ which is used to reduce the adjoint network of π_i:

$$(\boldsymbol{\mathcal{G}} + s\boldsymbol{\mathcal{C}})^T\boldsymbol{X}_a = \boldsymbol{d}_{\pi_i} \tag{4.21}$$

The reduced system resulting from applying MOR on (4.21) has the following form:

$$\hat{\boldsymbol{\mathcal{A}}}_a(s)\hat{\boldsymbol{X}}_a(s) = \hat{\boldsymbol{d}}_a \tag{4.22}$$

where $\hat{\boldsymbol{\mathcal{A}}}_a = \boldsymbol{\mathcal{Q}}^T \boldsymbol{\mathcal{A}}^T \boldsymbol{\mathcal{Q}}$ and $\hat{\boldsymbol{d}}_a = \boldsymbol{\mathcal{Q}}^T \boldsymbol{d}_{\pi_i}$.

Using $\boldsymbol{\mathcal{Q}}_a$ and $\tilde{\boldsymbol{X}}_a$, the sensitivity $\partial V / \partial \lambda_{i,j}$ results from the following equation:

$$\frac{\partial V}{\partial \lambda_{i,j}} = -\hat{\boldsymbol{X}}_a^T Q_a^T \left[\frac{\partial \boldsymbol{\mathcal{G}}_{\pi_i}}{\partial \lambda} + s \frac{\partial \boldsymbol{\mathcal{C}}_{\pi_i}}{\partial \lambda} \right] \boldsymbol{X}_{\pi_i}(s) \tag{4.23}$$

Therefore, to compute the sensitivity , one would have to solve the reduced system in (4.22) and then combine the obtained results with $\boldsymbol{\mathcal{Q}}_a$ as shown by (4.23). This approach would definitely provide the means to combine the concept of MOR with adjoint sensitivity analysis.

Nonetheless, this approach is valid only if the goal is to compute frequency-domain sensitivity for network π_i without having it connected to other networks, i.e. as a stand-alone network.

In addition, it would be a challenging task to generalize that approach to time-domain adjoint sensitivity analysis, as will be demonstrated in the following subsection.

4.2.2 Time-Domain Adjoint Sensitivity Analysis Using MOR

We now consider the task of adjoint sensitivity analysis on the entire network θ in the time domain. It should be stressed here that this time-domain approach would be the natural domain to tackle the issue of sensitivity analysis if the network θ contains nonlinear elements as initially assumed.

To illustrate the adjoint-based in tackling this issue, we assume that sensitivity information is required to optimize some cost (objective) function w.r.t. a set of design parameters. It should be noted that computing the sensitivity of some cost function is a more general framework from the one used earlier in Chapter 3 to describe the adjoint-based time-domain sensitivity, where the cost function was simply given by a single network variable and its waveform over a period of time. So, here instead of seeking the derivative of only one variable, we seek the derivative

of a cost function, which can include one or more variables as its arguments, over a period of time. To this end, we write this cost function as an integral of some instantaneous cost function as follows:

$$\psi(\lambda_{i,j}) = \int_0^{t^*} \gamma(\tilde{\boldsymbol{x}}(t), \lambda_{i,j})dt \tag{4.24}$$

where γ is an instantaneous known objective fuction, and $\{\lambda_{i,j}\}$ $(i = 1, \ldots, M, j = 1, \ldots, K_{\pi_i})$ refers to the set of all design parameters introduced earlier. The theory of adjoint variable method shows that those derivative can be computed through time-domain analysis of only two systems. The first system is the original system described by (4.7); analysis of this system is part of the simulation phase and its results are assumed available upon termination of the time-stepping algorithm. The other system needed to obtain sensitivity information is known as the adjoint system and is given by:

$$-\tilde{\boldsymbol{C}}^T \frac{d\tilde{\boldsymbol{x}}_a(t)}{dt} + \left(\tilde{\boldsymbol{G}}^T + \left(\frac{\partial \tilde{\boldsymbol{f}}}{\partial \tilde{\boldsymbol{x}}}\right)^T\right) \tilde{\boldsymbol{x}}_a(t) = -\nabla_{\tilde{\boldsymbol{x}}}\gamma \tag{4.25}$$

where $\nabla_{\tilde{\boldsymbol{x}}}\gamma$ is the gradient vector with respect to the $\tilde{\boldsymbol{x}}$. It can be shown that $\partial\psi/\partial\lambda_{i,j}$ is obtained from $\tilde{\boldsymbol{x}}(t)$ and $\tilde{\boldsymbol{x}}_a(t)$ through the following expression:

$$\begin{aligned}\frac{\partial \psi}{\partial \lambda_{i,j}} = & - \left(\tilde{\boldsymbol{x}}_a^T(t)\frac{\partial \tilde{\boldsymbol{x}}(t)}{\partial \lambda_{i,j}}\right)\Bigg|_{t=0} \\ & + \int_0^{t^*} \left(\frac{\partial \gamma}{\partial \lambda_{i,j}} + \tilde{\boldsymbol{x}}_a^T(t)\left[\frac{\partial \tilde{\boldsymbol{G}}}{\partial \lambda_{i,j}}\tilde{\boldsymbol{x}}(t) + \frac{\partial \tilde{\boldsymbol{C}}}{\partial \lambda_{i,j}}\frac{d\tilde{\boldsymbol{x}}(t)}{dt} + \frac{\partial \tilde{\boldsymbol{f}}(\tilde{\boldsymbol{x}})}{\partial \lambda_{i,j}}\right]\right)dt\end{aligned} \tag{4.26}$$

The above discussion demonstrates clearly that the objective of combining time-domain sensitivity analysis, based on the adjoint principal, with MOR is premised on the ability to compute the derivatives of the matrices $\tilde{\boldsymbol{C}}_{\pi_i}$ and $\tilde{\boldsymbol{G}}_{\pi_i}$ w.r.t. $\lambda_{i,j}$. These matrices, however, con-

tain the "stamps" contributed by the reduced systems which are obtained using the projection operation $\mathcal{Q}_{\pi_i}$.

Attempting to compute the derivative of the reduced system stamps necessitates computing the derivative of the projection operator $\mathcal{Q}_{\pi_i}$ as shown by (4.20).

4.3 Discussion

The idea of computing the derivative of the $\mathcal{Q}_{\pi_i}$ seems to be the only route that enables an adjoint approach to time-domain sensitivity, where reduced-order systems are present. The main goal of this thesis is to show that such a derivative can be computed by modifying original basis construction algorithm presented in Chapter 2 with little overhead in the computations.

Chapter 5

Adjoint Approach for Constructing Reduced Systems

This chapter addresses the hurdles in combining MOR with time-domain adjoint sensitivity described in the previous chapter. It also demonstrates the process of incorporating the adjoint techniques with MOR technique, the main contribution of this thesis.

5.1 Outline of the Proposed Approach

From earlier discussion, it is obvious that the main hurdle which seems to hinder applying time-domain adjoint sensitivity analysis to systems described by reduced-order models is centered around computing $\partial \mathcal{Q}_{\pi_i}/\partial \lambda_{i,j}$. This section addresses this issue by first taking a closer look at the Arnoldi process used to compute $\mathcal{Q}_{\pi_i}$ as this facilitates explaining the key ideas in the proposed approach. Section 5.3 then presents an overview of the proposed approach to computing $\partial \mathcal{Q}_{\pi_i}/\partial \lambda_{i,j}$.

5.2 Computing the Orthogonal Basis $\mathcal{Q}_{\pi_i}$

The main algorithmic procedure used in computing $\mathcal{Q}_{\pi_i}$ is based on using the Arnoldi algorithm. The Arnoldi algorithm was first introduced in Section 2.2.2 and Figure 2.5 described a pseudo-code-based implementation for systems with Single-Input Single-Output, i.e. single-port networks. For multi-port networks, however, a block Arnoldi version will be needed. Figures 5.1 and 5.2 depict the pseudo-code implementation of the block Arnoldi method. The main algorithm in Figure 5.1 runs the block version of the Arnoldi process (shown in Figure 2.5) which calls the ORTHOGONALIZE procedure (shown in Figure 5.2) to perform an orthogonalization using the Modified Gram Schmidt (MGS) [39] process on the input vectors. It is obvious from Figure 5.1 that $\mathcal{Q}_{\pi_i}$ is generated through an iterative process, in which each iteration involves the following computational procedures:

1. a matrix-vector multiplication,
2. solution of a linear system of equations,
3. a block orthogonalization, and finally
4. an MGS process.

As a result the dependence of $\mathcal{Q}_{\pi_i}$ on $\lambda_{i,j}$ become more complicated with every iteration.

5.3 Derivative Computation

The enabling idea behind the proposed approach lies mainly in the effect of the orthogonal basis $\mathcal{Q}_{\pi_i}$ after its generation. The key to achieve the derivative computation is found in the reduced system matrices for subnetwork π_i obtained using $\mathcal{Q}_{\pi_i}$, which are given by (4.4). Those matrices are known as the stamps of the subnetwork π_i in the reduced domain. It is to

Algorithm 3: Computing $\mathcal{Q}_{\pi_i}$

$\mathcal{G}_{\pi_i}, \mathcal{C}_{\pi_i}, \mathcal{B}_{\pi_i}, q$ **output**: $\mathcal{Q}_{\pi_i}$

$r \leftarrow \lceil \frac{q}{P_{\pi_i}} \rceil$

Solve: $\mathcal{G}_{\pi_i} \tilde{\mathcal{Q}}_0 = \mathcal{B}_{\pi_i}$

$\mathcal{Q}_0 \leftarrow$ ORTHOGONALIZE($\tilde{\mathcal{Q}}_0$)

begin

  for $m \leftarrow 1$ **to** $r-1$ **do**

    Solve: $\mathcal{G}_{\pi_i} \tilde{\mathcal{Q}}_m = -\mathcal{C}_{\pi_i} \mathcal{Q}_{m-1}$

    for $v = 0 \leftarrow$ **to** $m-1$ **do**

      $\tilde{\mathcal{Q}}_m \leftarrow \tilde{\mathcal{Q}}_m - \mathcal{Q}_v \left(\mathcal{Q}_v^T \tilde{\mathcal{Q}}_m \right)$

    $\mathcal{Q}_m \leftarrow$ ORTHOGONALIZE($\tilde{\mathcal{Q}}_m$)

  $\mathcal{Q}_{\pi_i} \leftarrow [\mathcal{Q}_0, \cdots, \mathcal{Q}_{r-1}]$

  Truncate: $\mathcal{Q}_{\pi_i}$ *to* q *columns only.*

  return $\mathcal{Q}_{\pi_i}$

end

Figure 5.1: Pseudo-code description for the Arnoldi process

Procedure : ORTHOGONALIZE

input: A set of vectors $[\boldsymbol{v}_i] \in \mathbb{R}^L$, $i = 0, \cdots, l-1$

output: An orthogonal basis $\mathcal{Q}$ for $[\boldsymbol{v}_i]$

$\boldsymbol{q}_0 \leftarrow \boldsymbol{v}_0 / \|\boldsymbol{v}_0\|$

begin

  for $k \leftarrow 1$ **to** $l-1$ **do**

    $\tilde{\boldsymbol{q}}_k \leftarrow \boldsymbol{v}_k$

    for $h \leftarrow 0$ **to** $k-1$ **do**

      $\tilde{\boldsymbol{q}}_k \leftarrow \tilde{\boldsymbol{q}}_k - \left(\tilde{\boldsymbol{q}}_k^T \boldsymbol{q}_h \right) \boldsymbol{q}_h$

    $\boldsymbol{q}_k \leftarrow \tilde{\boldsymbol{q}}_k / \|\tilde{\boldsymbol{q}}_k\|$

  $\mathcal{Q} \leftarrow [\boldsymbol{q}_0, \cdots \boldsymbol{q}_{l-1}]$

  return $\mathcal{Q}$

end

Figure 5.2: Pseudo-code description for the Arnoldi process (*Cont.*). The ORTHOGONALIZE procedure is given based on the Modified Gram Schmidt process.

be noted that computing the derivatives of any of those stamps with respect $\lambda_{i,j}$ represents the objective that enables applying time-domain adjoint technique. Without loss of generality, we consider the stamp of the reduced matrix $\hat{\boldsymbol{\mathcal{G}}}_{\pi_i}$, and show how to obtain its derivative, where the results thus established can be easily extended to computing the derivatives of $\hat{\boldsymbol{\mathcal{C}}}_{\pi_i}$ and $\hat{\boldsymbol{\mathcal{B}}}_{\pi_i}$. We denote by $\left[\hat{\boldsymbol{\mathcal{G}}}_{\pi_i}\right]_{m,n}$ the (m,n) block entry of $\hat{\boldsymbol{\mathcal{G}}}_{\pi_i}$ obtained from

$$\left[\hat{\boldsymbol{\mathcal{G}}}_{\pi_i}\right]_{m,n} = \boldsymbol{\mathcal{Q}}_{\pi_i}^{(m)T} \boldsymbol{\mathcal{G}}_{\pi_i} \boldsymbol{\mathcal{Q}}_{\pi_i}^{(n)} \tag{5.1}$$

where $\boldsymbol{\mathcal{Q}}_{\pi_i}^{(m)}$ and $\boldsymbol{\mathcal{Q}}_{\pi_i}^{(n)}$ are the block columns of $\boldsymbol{\mathcal{Q}}_{\pi_i}$ generated after the m^{th} and n^{th} iteration of the outer loop in Figure 5.1, respectively. The derivative of the (m,n) block can be obtained using the chain rule of differentiation as follows,

$$\frac{\partial}{\partial \lambda_{i,j}}\left[\hat{\boldsymbol{\mathcal{G}}}_{\pi_i}\right]_{m,n} = \left(\frac{\partial \boldsymbol{\mathcal{Q}}_{\pi_i}^{(m)}}{\partial \lambda_{i,j}}\right)^T \boldsymbol{\mathcal{G}}_{\pi_i} \boldsymbol{\mathcal{Q}}_{\pi_i}^{(n)} + \boldsymbol{\mathcal{Q}}_{\pi_i}^{(m)T} \frac{\partial \boldsymbol{\mathcal{G}}_{\pi_i}}{\partial \lambda_{i,j}} \boldsymbol{\mathcal{Q}}_{\pi_i}^{(n)} + \boldsymbol{\mathcal{Q}}_{\pi_i}^{(m)T} \boldsymbol{\mathcal{G}}_{\pi_i} \left(\frac{\partial \boldsymbol{\mathcal{Q}}_{\pi_i}^{(n)}}{\partial \lambda_{i,j}}\right) \tag{5.2}$$

Our approach to demonstrate how to compute $\partial \boldsymbol{\mathcal{Q}}_{\pi_i} / \partial \lambda_{i,j}$ is based on the principal of mathematical induction, where we prove that $\partial \boldsymbol{\mathcal{Q}}_{\pi_i}^{(m)} / \partial \lambda_{i,j}$ can be computed if $\partial \boldsymbol{\mathcal{Q}}_{\pi_i}^{(m-1)} / \partial \lambda_{i,j}$ is readily available. We then proceed to show how to compute $\partial \boldsymbol{\mathcal{Q}}_{\pi_i}^{(0)} / \partial \lambda_{i,j}$.

We note from the descriptive pseudo-code in Figure 5.1 that the matrix $\boldsymbol{\mathcal{Q}}_{\pi_i}^{(m)}$ is the output of the ORTHOGONALIZE procedure, with the matrix $\tilde{\boldsymbol{\mathcal{Q}}}_m$ being the supplied input parameter. In that sense, ORTHOGONALIZE functions as a matrix-to-matrix mapping. This notion can be expressed formally by denoting this mapping by $\boldsymbol{\Upsilon}$, where $\boldsymbol{\Upsilon} : \mathbb{R}^{N_{\pi_i} \times P_{\pi_i}} \rightarrow \mathbb{R}^{N_{\pi_i} \times P_{\pi_i}}$, and therefore

$$\boldsymbol{\mathcal{Q}}_{\pi_i}^{(m)} = \boldsymbol{\Upsilon}\left(\tilde{\boldsymbol{\mathcal{Q}}}_m\right) \tag{5.3}$$

We partition the mapping $\boldsymbol{\Upsilon}$ into a set of P_{π_i} column-wise mappings,

$$\boldsymbol{\Upsilon} \equiv \left[\boldsymbol{\Upsilon}_1 \boldsymbol{\Upsilon}_2 \cdots \boldsymbol{\Upsilon}_{P_{\pi_i}}\right] \tag{5.4}$$

where $\boldsymbol{\Upsilon}_k : \mathbb{R}^{N_{\pi_i} \times P_{\pi_i}} \rightarrow \mathbb{R}^{N_{\pi_i} \times 1}$, defined by

$$\boldsymbol{\mathcal{Q}}_{\pi_i}^{(m)}(k) = \boldsymbol{\Upsilon}_k \left(\tilde{\boldsymbol{\mathcal{Q}}}_m \right) \tag{5.5}$$

and hence,

$$\frac{\partial \boldsymbol{\mathcal{Q}}_{\pi_i}^{(m)}(k)}{\partial \lambda_{i,j}} = \frac{\partial \boldsymbol{\Upsilon}_k \left(\tilde{\boldsymbol{\mathcal{Q}}}_m \right)}{\partial \lambda_{i,j}} \tag{5.6}$$

with $\boldsymbol{\mathcal{Q}}_{\pi_i}^{(m)}(k)$ denoting the k^{th} column in $\boldsymbol{\mathcal{Q}}_{\pi_i}^{(m)}$. In fact, $\boldsymbol{\Upsilon}_k$ results from the k^{th} step in the outer loop of the MGS process (Figure 5.2), and may be expressed in the following closed-form relation

$$\boldsymbol{\Upsilon}_k \left(\tilde{\boldsymbol{\mathcal{Q}}}_m \right) = \frac{\tilde{\boldsymbol{\mathcal{Q}}}_m(k) - \sum_{h=0}^{k-1} \left\langle \tilde{\boldsymbol{\mathcal{Q}}}_m(k), \boldsymbol{\mathcal{Q}}_{\pi_i}^{(h)} \right\rangle \boldsymbol{\mathcal{Q}}_{\pi_i}^{(h)}}{\left\| \tilde{\boldsymbol{\mathcal{Q}}}_m(k) - \sum_{h=0}^{k-1} \left\langle \tilde{\boldsymbol{\mathcal{Q}}}_m(k), \boldsymbol{\mathcal{Q}}_{\pi_i}^{(h)} \right\rangle \boldsymbol{\mathcal{Q}}_{\pi_i}^{(h)} \right\|} \tag{5.7}$$

or equivalently using the notation introduced in (5.5)

$$\boldsymbol{\Upsilon}_k \left(\tilde{\boldsymbol{\mathcal{Q}}}_m \right) = \frac{\tilde{\boldsymbol{\mathcal{Q}}}_m(k) - \sum_{h=0}^{k-1} \left\langle \tilde{\boldsymbol{\mathcal{Q}}}_m(k), \boldsymbol{\Upsilon}_h \right\rangle \boldsymbol{\Upsilon}_h}{\left\| \tilde{\boldsymbol{\mathcal{Q}}}_m(k) - \sum_{h=0}^{k-1} \left\langle \tilde{\boldsymbol{\mathcal{Q}}}_m(k), \boldsymbol{\Upsilon}_h \right\rangle \boldsymbol{\Upsilon}_h \right\|} \tag{5.8}$$

where $\langle \cdot, \cdot \rangle$ and $\| \cdot \|$ are the inner-product and norm mappings defined on $\mathbb{R}^{N_{\pi_i}}$, respectively, and $\tilde{\boldsymbol{\mathcal{Q}}}_m(k)$ is the k^{th} column of $\tilde{\boldsymbol{\mathcal{Q}}}_m$, which is passed as the input parameter. We now seek to compute $\partial \boldsymbol{\mathcal{Q}}_{\pi_i}^{(m)} / \partial \lambda_{i,j}$ by computing the derivative of its columns, $\boldsymbol{\mathcal{Q}}_{\pi_i}^{(m)}(k)$, individually, through finding the derivative of $\boldsymbol{\Upsilon}_k$. Note that $\boldsymbol{\Upsilon}_k$ can be cast in the following form

$$\boldsymbol{\Upsilon}_k \left(\tilde{\boldsymbol{\mathcal{Q}}}_m \right) = \frac{\boldsymbol{\chi}_k}{\| \boldsymbol{\chi}_k \|} \tag{5.9}$$

where

$$\boldsymbol{\chi}_k = \tilde{\boldsymbol{\mathcal{Q}}}_m(k) - \sum_{h=0}^{k-1} \left\langle \tilde{\boldsymbol{\mathcal{Q}}}_m(k), \boldsymbol{\Upsilon}_h \right\rangle \boldsymbol{\Upsilon}_h \tag{5.10}$$

Differentiating both sides of (5.9) w.r.t. $\lambda_{i,j}$ yields

$$\frac{\partial \boldsymbol{\Upsilon}_k}{\partial \lambda_{i,j}} = \frac{1}{\langle \boldsymbol{\chi}_k, \boldsymbol{\chi}_k \rangle} \left(\|\boldsymbol{\chi}_k\| \frac{\partial \boldsymbol{\chi}_k}{\partial \lambda_{i,j}} - \left\langle \frac{\partial \boldsymbol{\chi}_k}{\partial \lambda_{i,j}}, \boldsymbol{\chi}_k \right\rangle \frac{\boldsymbol{\chi}_k}{\|\boldsymbol{\chi}_k\|} \right) \tag{5.11}$$

From (5.11), it is straightforward to conclude that it is possible to compute $\frac{\partial \boldsymbol{\Upsilon}_k}{\partial \lambda_{i,j}}$ if $\frac{\partial \boldsymbol{\chi}_k}{\partial \lambda_{i,j}}$ is available. $\frac{\partial \boldsymbol{\chi}_k}{\partial \lambda_{i,j}}$ can be computed from (5.10) and follows from,

$$\begin{aligned} \frac{\partial \boldsymbol{\chi}_k}{\partial \lambda_{i,j}} = & \quad \frac{\partial \tilde{\boldsymbol{\mathcal{Q}}}_m(k)}{\partial \lambda_{i,j}} - \sum_{h=0}^{k-1} \left\langle \frac{\partial \tilde{\boldsymbol{\mathcal{Q}}}_m(k)}{\partial \lambda_{i,j}}, \boldsymbol{\Upsilon}_h \right\rangle \boldsymbol{\Upsilon}_h \\ & + \left\langle \tilde{\boldsymbol{\mathcal{Q}}}_m(k), \frac{\partial \boldsymbol{\Upsilon}_h}{\partial \lambda_{i,j}} \right\rangle \boldsymbol{\Upsilon}_h + \left\langle \tilde{\boldsymbol{\mathcal{Q}}}_m(k), \boldsymbol{\Upsilon}_h \right\rangle \frac{\partial \boldsymbol{\Upsilon}_h}{\partial \lambda_{i,j}} \end{aligned} \tag{5.12}$$

Examining (5.11) and the right-side of (5.12) shows, as noted previously, that $\frac{\partial \boldsymbol{\chi}_k}{\partial \lambda_{i,j}}$ can be deduced from the knowledge of $\partial \tilde{\boldsymbol{\mathcal{Q}}}_m(k)/\partial \lambda_{i,j}$ and previously calculated $\partial \boldsymbol{\Upsilon}_h/\partial \lambda_{i,j}$, $(h < k)$. Therefore, to complete this part, we only need to show how to evaluate the derivative of the first column $\left(\frac{\partial \boldsymbol{\Upsilon}_0}{\partial \lambda_{i,j}}\right)$, and postpone dealing with $\partial \tilde{\boldsymbol{\mathcal{Q}}}_m(k)/\partial \lambda_{i,j}$ to the next subsection. Evaluating $\left(\frac{\partial \boldsymbol{\Upsilon}_0}{\partial \lambda_{i,j}}\right)$ follows easily, by noting that $\boldsymbol{\Upsilon}_0$ results from step 1, and hence can be described by

$$\boldsymbol{\Upsilon}_0 = \frac{\tilde{\boldsymbol{\mathcal{Q}}}_m(0)}{\|\tilde{\boldsymbol{\mathcal{Q}}}_m(0)\|} \tag{5.13}$$

and, therefore, its derivative can be obtained from

$$\begin{aligned} \frac{\partial \boldsymbol{\Upsilon}_0}{\partial \lambda_{i,j}} = & \quad \frac{1}{\left\langle \tilde{\boldsymbol{\mathcal{Q}}}_m(0), \tilde{\boldsymbol{\mathcal{Q}}}_m(0) \right\rangle} \\ & \times \left(\left\| \tilde{\boldsymbol{\mathcal{Q}}}_m(0) \right\| \frac{\partial \tilde{\boldsymbol{\mathcal{Q}}}_m(0)}{\partial \lambda_{i,j}} - \left\langle \frac{\partial \tilde{\boldsymbol{\mathcal{Q}}}_m(0)}{\partial \lambda_{i,j}}, \tilde{\boldsymbol{\mathcal{Q}}}_m(0) \right\rangle \frac{\tilde{\boldsymbol{\mathcal{Q}}}_m(0)}{\left\| \tilde{\boldsymbol{\mathcal{Q}}}_m(0) \right\|} \right) \end{aligned} \tag{5.14}$$

From (5.6), (5.11), (5.12), and (5.14), it can be easily concluded that computing $\partial \boldsymbol{\mathcal{Q}}_{\pi_i}^{(m)} / \partial \lambda_{i,j}$ is possible if one can evaluate the columns of the matrix $\partial \tilde{\boldsymbol{\mathcal{Q}}}_m / \partial \lambda_{i,j}$. This issue is discussed next.

5.3.1 Computing $\partial \tilde{\boldsymbol{\mathcal{Q}}}_m / \partial \lambda_{i,j}$

From figure 5.1, recall that $\tilde{\boldsymbol{\mathcal{Q}}}_m$ is the matrix argument passed to the ORTHOGONALIZE procedure, and $\tilde{\boldsymbol{\mathcal{Q}}}_m(k)$ is its k^{th} column. Note also that $\tilde{\boldsymbol{\mathcal{Q}}}_m$ is sent to this procedure upon the completion of the m^{th} iteration. In fact, $\tilde{\boldsymbol{\mathcal{Q}}}_m$ is the outcome of running a block version of the MGS procedure for m times with starting matrix $-\boldsymbol{\mathcal{G}}_{\pi_i}^{-1} \boldsymbol{\mathcal{C}}_{\pi_i} \boldsymbol{\mathcal{Q}}_{\pi_i}^{(m-1)}$. Hence, one can use the MGS formulation to write the following closed form expression for $\tilde{\boldsymbol{\mathcal{Q}}}_m$,

$$\tilde{\boldsymbol{\mathcal{Q}}}_m = -\boldsymbol{\mathcal{G}}_{\pi_i}^{-1} \boldsymbol{\mathcal{C}}_{\pi_i} \boldsymbol{\mathcal{Q}}_{\pi_i}^{(m-1)} - \sum_{v=0}^{m-1} \boldsymbol{\mathcal{Q}}_{\pi_i}^{(v)} \left\langle -\boldsymbol{\mathcal{G}}_{\pi_i}^{-1} \boldsymbol{\mathcal{C}}_{\pi_i} \boldsymbol{\mathcal{Q}}_{\pi_i}^{(m-1)}, \boldsymbol{\mathcal{Q}}_{\pi_i}^{(v)} \right\rangle \tag{5.15}$$

Thus differentiating both sides of (5.15) yields

$$\begin{aligned} \frac{\partial \tilde{\boldsymbol{\mathcal{Q}}}_m}{\partial \lambda_{i,j}} = \frac{\partial \boldsymbol{\xi}_{m-1}}{\partial \lambda_{i,j}} - \sum_{v=0}^{m-1} \quad & \left(\boldsymbol{\mathcal{Q}}_{\pi_i}^{(v)} \left\langle \frac{\partial \boldsymbol{\xi}_{m-1}}{\partial \lambda_{i,j}}, \boldsymbol{\mathcal{Q}}_{\pi_i}^{(v)} \right\rangle \right. \\ & + \boldsymbol{\mathcal{Q}}_{\pi_i}^{(v)} \left\langle \boldsymbol{\xi}_{m-1}, \frac{\partial \boldsymbol{\mathcal{Q}}_{\pi_i}^{(v)}}{\partial \lambda_{i,j}} \right\rangle \\ & \left. + \frac{\partial \boldsymbol{\mathcal{Q}}_{\pi_i}^{(v)}}{\partial \lambda_{i,j}} \left\langle \boldsymbol{\xi}_{m-1}, \boldsymbol{\mathcal{Q}}_{\pi_i}^{(v)} \right\rangle \right) \end{aligned} \tag{5.16}$$

where

$$\boldsymbol{\xi}_{m-1} = -\boldsymbol{\mathcal{G}}_{\pi_i}^{-1} \boldsymbol{\mathcal{C}}_{\pi_i} \boldsymbol{\mathcal{Q}}_{\pi_i}^{(m-1)} \tag{5.17}$$

$\partial \boldsymbol{\xi}_{m-1}/\partial \lambda_{i,j}$ in (5.16) can be computed from (5.17) by differentiating both sides (after post multiplication by $\boldsymbol{\mathcal{G}}_{\pi_i}$)

$$\frac{\partial \boldsymbol{\mathcal{G}}_{\pi_i}}{\partial \lambda_{i,j}} \boldsymbol{\xi}_{m-1} + \boldsymbol{\mathcal{G}}_{\pi_i} \frac{\partial \boldsymbol{\xi}_{m-1}}{\partial \lambda_{i,j}} = -\frac{\partial \boldsymbol{\mathcal{C}}_{\pi_i}}{\partial \lambda_{i,j}} \boldsymbol{\mathcal{Q}}_{\pi_i}^{(m-1)} - \boldsymbol{\mathcal{C}}_{\pi_i} \frac{\partial \boldsymbol{\mathcal{Q}}_{\pi_i}^{(m-1)}}{\partial \lambda_{i,j}} \tag{5.18}$$

and therefore,

$$\frac{\partial \boldsymbol{\xi}_{m-1}}{\partial \lambda_{i,j}} = -\boldsymbol{\mathcal{G}}_{\pi_i}^{-1} \left(\frac{\partial \boldsymbol{\mathcal{C}}_{\pi_i}}{\partial \lambda_{i,j}} \boldsymbol{\mathcal{Q}}_{\pi_i}^{(m-1)} + \boldsymbol{\mathcal{C}}_{\pi_i} \frac{\partial \boldsymbol{\mathcal{Q}}_{\pi_i}^{(m-1)}}{\partial \lambda_{i,j}} + \frac{\partial \boldsymbol{\mathcal{G}}_{\pi_i}}{\partial \lambda_{i,j}} \boldsymbol{\xi}_{m-1} \right) \tag{5.19}$$

5.3.2 Summary

The above derivations can be summed up as follows. Computing $\partial \boldsymbol{\mathcal{Q}}_{\pi_i}^{(m)}/\partial \lambda_{i,j}$ is feasible if and only if $\partial \boldsymbol{\mathcal{Q}}_{\pi_i}^{(m-1)}/\partial \lambda_{i,j}$ can be computed. To conclude the induction argument, we only need to show how to compute $\partial \boldsymbol{\mathcal{Q}}_{\pi_i}^{(0)}/\partial \lambda_{i,j}$. This calculation follows directly by observing from step 3 in Figure 5.1, that $\boldsymbol{\mathcal{Q}}_{\pi_i}^{(0)}$ results from applying the mapping associated with the ORTHOGONALIZE procedure on $\tilde{\boldsymbol{\mathcal{Q}}}_0$. Based on the above derivations, $\partial \boldsymbol{\mathcal{Q}}_{\pi_i}^{(0)}/\partial \lambda_{i,j}$ can be computed if $\partial \tilde{\boldsymbol{\mathcal{Q}}}_0/\partial \lambda_{i,j}$ is available. The latter can be easily obtained by noting from step 2 in Figure 5.1, that it essentially results from solving the system $\boldsymbol{\mathcal{G}}_{\pi_i} \tilde{\boldsymbol{\mathcal{Q}}}_0 = \boldsymbol{\mathcal{B}}_{\pi_i}$. Hence, $\partial \tilde{\boldsymbol{\mathcal{Q}}}_0/\partial \lambda_{i,j}$ can be obtained by solving the following system

$$\boldsymbol{\mathcal{G}}_{\pi_i} \frac{\partial \tilde{\boldsymbol{\mathcal{Q}}}_0}{\partial \lambda_{i,j}} = -\frac{\partial \boldsymbol{\mathcal{G}}_{\pi_i}}{\partial \lambda_{i,j}} \tilde{\boldsymbol{\mathcal{Q}}}_0 \tag{5.20}$$

5.4 Implementation of the Proposed Algorithm

This section builds on the derivations presented in the previous section to establish a computational procedure that can be implemented on top of the original basis generation algorithm. The basic objective of the implementation presented next is to have the derivative of the or-

thogonal basis computed with minimal computational overhead. This objective will be carried out by utilizing the results of the intermediate steps in the original algorithm. Another advantage that will result of that, is that implementation of the proposed algorithm will lend itself more naturally to parallel implementation.

The main computational steps needed to find $\partial \boldsymbol{\mathcal{Q}}_{\pi_i} / \partial \lambda_{i,j}$ are those shown in (5.11), (5.12), and (5.16). To show how these equations can be implemented, we first simplify the notation used therein to make the derivations easy to follow. We use $\tilde{\boldsymbol{\mathcal{Q}}}_{\lambda,m}$ to denote $\partial \tilde{\boldsymbol{\mathcal{Q}}}_m / \partial \lambda_{i,j}$. We will also drop the subscript π_i from $\boldsymbol{\mathcal{Q}}_{\pi_i}$, and use m in the subscript letter to denote its m^{th} block. With this new notation, (5.16) may be implemented by first using the following initialization

$$\tilde{\boldsymbol{\mathcal{Q}}}_{\lambda,m} \leftarrow \frac{\partial \boldsymbol{\xi}_{m-1}}{\partial \lambda_{i,j}}$$

and then through using the update procedure shown in Figure 5.3.

for $v \leftarrow 0$ **to** $m-1$ **do**

$$\tilde{\boldsymbol{\mathcal{Q}}}_{\lambda,m} \leftarrow \tilde{\boldsymbol{\mathcal{Q}}}_{\lambda,m} - \left(\boldsymbol{\mathcal{Q}}_v \left\langle \frac{\partial \boldsymbol{\xi}_{m-1}}{\partial \lambda_{i,j}}, \boldsymbol{\mathcal{Q}}_v \right\rangle + \boldsymbol{\mathcal{Q}}_v \left\langle \boldsymbol{\xi}_{m-1}, \frac{\partial \boldsymbol{\mathcal{Q}}_v}{\partial \lambda_{i,j}} \right\rangle + \frac{\partial \boldsymbol{\mathcal{Q}}_v}{\partial \lambda} \langle \boldsymbol{\xi}_{m-1}, \boldsymbol{\mathcal{Q}}_v \rangle \right)$$

Figure 5.3: An update procedure implementing (5.16)

An alternative update procedure, however, can be derived through utilizing the following

initialization

$$\begin{aligned} \tilde{\mathcal{Q}}_m &\leftarrow \boldsymbol{\xi}_{m-1} \\ \tilde{\mathcal{Q}}_{\lambda,m} &\leftarrow \frac{\partial \boldsymbol{\xi}_{m-1}}{\partial \lambda_{i,j}} \end{aligned}$$

and the update procedure shown in Figure 5.4.

for $v \leftarrow 0$ **to** $m-1$ **do**

$$\begin{aligned} \tilde{\mathcal{Q}}_{\lambda,m} \leftarrow \tilde{\mathcal{Q}}_{\lambda,m} - \Big(& \mathcal{Q}_v \left\langle \tilde{\mathcal{Q}}_{\lambda,m}, \mathcal{Q}_v \right\rangle \\ & + \mathcal{Q}_v \left\langle \tilde{\mathcal{Q}}_m, \frac{\partial \mathcal{Q}_v}{\partial \lambda_{i,j}} \right\rangle \\ & + \frac{\partial \mathcal{Q}_v}{\partial \lambda} \left\langle \tilde{\mathcal{Q}}_m, \mathcal{Q}_v \right\rangle \Big) \end{aligned}$$

$$\tilde{\mathcal{Q}}_m \leftarrow \tilde{\mathcal{Q}}_m - \mathcal{Q}_v \left(\tilde{\mathcal{Q}}_v^T \mathcal{Q} \right)$$

Figure 5.4: A *modified* update procedure implementing (5.16)

A proof of the equivalence between the two procedures in Figure 5.3 and Figure 5.4 is given in Appendix B. Although the two update procedures are equivalent, the one shown in Figure 5.3 utilizes the Classical Gram Schmidt (CGS) process in updating $\partial \tilde{\mathcal{Q}} / \partial \lambda_{i,j}$, while the one shown in Figure 5.4 utilizes the Modified Gram Schmidt (MGS) process in the updates. Numerically, it has been shown that MGS offers better numerical accuracy than the CGS [39].

5.4.1 Procedural Description of the Proposed Algorithm

Figures 5.5-5.7 show a procedural description of the proposed algorithm in the form of a pseudo-code. Figure 5.5 shows the main body of the proposed algorithm, which calls a modified version of the ORTHOGONALIZE procedure. This version is shown in Figure 5.6, and is

termed MODIFIED_ORTHOGONALIZE. It is worth noting that implementation of the proposed derivative computation algorithm is cast as additional components added on top of the original basis computation algorithm of Figures 5.1 and 5.2. This will also be useful in analyzing the computational complexity in the following section. It is also to be noted that an extra procedural component (NORM_DERIVATIVE) needed in the algorithm is shown in Figure 5.7. This procedure is based on the expression in (5.11) used in computing the derivative of a normalized vector. The steps shown in Framed boxes in Figures 5.5 and 5.6 denote those steps needed to compute $\partial \mathcal{Q} / \partial \lambda_{i,j}$.

5.5 Computational Complexity

This part presents a brief analysis for the extra computational cost needed to compute the derivative of the projection matrix. The main computational overhead can be seen in lines 4, 8, and 10 of Figure 5.5, in which a $\lceil \frac{q}{P_{\pi_i}} \rceil$ Forward/Backward (F/B) substitutions (using the L/U factors of G_{π_i}) and 9 matrix-matrix multiplications are performed per a single iteration. Given that the matrices G_{π_i} and C_{π_i} (and their derivatives) are typically sparse with approximately $\mathrm{O}(N_{\pi_i})$ nonzero entries, the total additional computations will be $\mathrm{O}(N_{\pi_i})$ for the F/B substitutions and $3N_{\pi_i}P_{\pi_i}\,(1 + 2P_{\pi_i})$ for the matrix-matrix multiplications. This makes the additional cost approximately $\mathrm{O}\left(N_{\pi_i}{P_{\pi_i}}^2\right)$ per single iteration of the Modified-Arnoldi process, or $\mathrm{O}\left(N_{\pi_i}{P_{\pi_i}}^2\lceil \frac{q}{P_{\pi_i}} \rceil\right)$ for the whole process. Therefore, the additional cost scales linearly with the large size of the original subnetwork π_i, and is performed only once. It is to be noted that in classical adjoint approaches to time-domain sensitivity analysis without MOR, the full size of the subnetwork will have to be incorporated in the formulation of (4.7), thus requiring a computational cost of at least $\mathrm{O}\left({N_{\pi_i}}^{\alpha}\right)$, $1.1 < \alpha < 1.5$ [5], at each time step during the time-stepping scheme.

Algorithm 5: Computing $\mathcal{Q}_{\pi_i}$ and $\partial\mathcal{Q}/\partial\lambda_{i,j}$

input: $\mathcal{G}_{\pi_i}, \mathcal{C}_{\pi_i}, \mathcal{B}_{\pi_i}, \frac{\partial\mathcal{G}_{\pi_i}}{\partial\lambda_{i,j}}, \frac{\partial\mathcal{C}_{\pi_i}}{\partial\lambda_{i,j}}, r$

output: $\mathcal{Q}_{\pi_i}, \frac{\partial\mathcal{Q}_{\pi_i}}{\partial\lambda_{i,j}}$

/*This algorithm computes an orthogonal basis for the Krylov subspace $\mathrm{K}_r\left(\mathcal{G}_{\pi_i}^{-1}\mathcal{C}_{\pi_i}, \mathcal{G}_{\pi_i}^{-1}\mathcal{B}_{\pi_i}\right)$. It also returns the derivative of this orthogonal basis w.r.t. $\lambda_{i,j}$, given $\frac{\partial\mathcal{G}_{\pi_i}}{\partial\lambda_{i,j}}$, $\frac{\partial\mathcal{C}_{\pi_i}}{\partial\lambda_{i,j}}$. */

begin

  $r \leftarrow \lceil \frac{q}{P_{\pi_i}} \rceil$

  Solve: $\mathcal{G}_{\pi_v}\tilde{\mathcal{Q}}_0 = \mathcal{B}_{\pi_i}$

  Solve: $\mathcal{G}_{\pi_i}\tilde{\mathcal{Q}}_{\lambda,0} = \frac{\partial\mathcal{G}_{\pi_i}}{\partial\lambda_{i,j}}\tilde{\mathcal{Q}}_0$

  $[\mathcal{Q}_0, \mathcal{Q}_{\lambda,0}] \leftarrow \textsc{Modified\_Orthogonalize}\left(\tilde{\mathcal{Q}}_0, \tilde{\mathcal{Q}}_{\lambda,0}\right)$

  for $m \leftarrow 1$ **to** $r-1$ **do**

    Solve: $\mathcal{G}_{\pi_i}\tilde{\mathcal{Q}}_m = -\mathcal{C}_{\pi_i}\mathcal{Q}_{m-1}$

    Solve: $\mathcal{G}_{\pi_i}\tilde{\mathcal{Q}}_{\lambda,m} = -\frac{\partial\mathcal{C}_{\pi_i}}{\partial\lambda_{i,j}}\mathcal{Q}_{m-1} - \mathcal{C}_{\pi_i}\mathcal{Q}_{\lambda,m-1} - \frac{\partial\mathcal{G}_{\pi_i}}{\partial\lambda_{i,j}}\tilde{\mathcal{Q}}_m$

    for $v \leftarrow 0$ **to** $m-1$ **do**

      $\tilde{\mathcal{Q}}_{\lambda,m} \leftarrow \tilde{\mathcal{Q}}_{\lambda,m} - \left[\mathcal{Q}_v\left(\tilde{\mathcal{Q}}_{\lambda,m}^T\mathcal{Q}_v\right) + \mathcal{Q}_v\left(\tilde{\mathcal{Q}}_m^T\mathcal{Q}_{\lambda,v}\right) + \mathcal{Q}_{\lambda,v}\left(\tilde{\mathcal{Q}}_m^T\mathcal{Q}_v\right)\right]$

      $\tilde{\mathcal{Q}}_m \leftarrow \tilde{\mathcal{Q}}_m - \mathcal{Q}_v\left(\mathcal{Q}_v^T\tilde{\mathcal{Q}}_m\right)$

    $[\mathcal{Q}_m, \mathcal{Q}_{\lambda,m}] \leftarrow \textsc{Modified\_Orthogonalize}\left(\tilde{\mathcal{Q}}_m, \tilde{\mathcal{Q}}_{\lambda,m}\right)$

  $\mathcal{Q}_{\pi_i} \leftarrow [\mathcal{Q}_0, \cdots, \mathcal{Q}_{r-1}]$

  $\frac{\partial\mathcal{Q}_{\pi_i}}{\partial\lambda_{i,j}} \leftarrow [\mathcal{Q}_{\lambda,0}, \cdots, \mathcal{Q}_{\lambda,r-1}]$

  Truncate: $\mathcal{Q}_{\pi_i}$ *and* $\frac{\partial\mathcal{Q}_{\pi_i}}{\partial\lambda_{i,j}}$ *to* q *columns only.*

  return $\left(\mathcal{Q}_{\pi_i}, \frac{\partial\mathcal{Q}_{\pi_i}}{\partial\lambda_{i,j}}\right)$

end

Figure 5.5: Pseudo-code description for the proposed Algorithm

Procedure : MODIFIED_ORTHOGONALIZE

input: $\tilde{\mathcal{Q}}_i, \frac{\partial \tilde{\mathcal{Q}}_i}{\partial \lambda}, i = 0, \cdots, l-1$
output: $\mathcal{Q}, \frac{\partial \mathcal{Q}}{\partial \lambda}$
/*Given a set of vectors $\tilde{\mathcal{Q}}_i$ and their derivative w.r.t. an arbitrary parameter λ, this procedure computes an orthogonal basis for the subspace spanned by $\tilde{\mathcal{Q}}_i$. It also returns the derivative of this basis w.r.t. λ. */

1 **begin**
2 $\quad q_0 \leftarrow \tilde{\mathcal{Q}}_0 / \|\tilde{\mathcal{Q}}_0\|$
3 $\quad \boxed{q_{\lambda,0} \leftarrow \text{NORM\_DERIVATIVE}\left(\tilde{\mathcal{Q}}_0, \frac{\partial \tilde{\mathcal{Q}}_0}{\partial \lambda}\right)}$
4 $\quad$ **for** $k = 1 \leftarrow$ **to** $l-1$ **do**
5 $\quad\quad \tilde{q}_k \leftarrow \tilde{\mathcal{Q}}_k$
6 $\quad\quad \boxed{\tilde{q}_{\lambda,k} \leftarrow \frac{\partial \tilde{\mathcal{Q}}_k}{\partial \lambda}}$
7 $\quad\quad$ **for** $h = 0 \leftarrow$ **to** $k-1$ **do**
8 $\quad\quad\quad \boxed{\tilde{q}_{\lambda,k} \leftarrow \tilde{q}_{\lambda,k} - \left(\left(\tilde{q}_{\lambda,k}^T q_h\right) q_h + \left(\tilde{q}_k^T q_{\lambda,h}\right) q_h + \left(\tilde{q}_k^T q_h\right) q_{\lambda,h} \right)}$
9 $\quad\quad\quad \tilde{q}_k \leftarrow \tilde{q}_k - \left(\tilde{q}_k^T q_h\right) q_h$
10 $\quad\quad q_k \leftarrow \tilde{q}_k / \|\tilde{q}_k\|$
11 $\quad\quad \boxed{q_{\lambda,k} \leftarrow \text{NORM\_DERIVATIVE}\left(\tilde{q}_k, \tilde{q}_{\lambda,k}\right)}$
12 $\quad \mathcal{Q} \leftarrow \left[q_0, \cdots, q_{l-1}\right]$
13 $\quad \frac{\partial \mathcal{Q}}{\partial \lambda} \leftarrow \left[q_{\lambda,0}, \cdots, q_{\lambda,l-1}\right]$
14 $\quad$ **return** $\left(\mathcal{Q}, \frac{\partial \mathcal{Q}}{\partial \lambda}\right)$
15 **end**

Figure 5.6: Pseudo-code description for the proposed algorithm (*cont.*)

Procedure : NORM_DERIVATIVE

input: $\tilde{\boldsymbol{q}}$, $\tilde{\boldsymbol{q}}_\lambda$

output: $\boldsymbol{q}_\lambda$

/*Given a vector $\boldsymbol{q}$ and its derivative w.r.t. an arbitrary parameter λ, this procedure returns the derivative of its normalized vector, $\boldsymbol{q}$, (defined by $\boldsymbol{q} = \tilde{\boldsymbol{q}}/\|\tilde{\boldsymbol{q}}\|$ w.r.t. λ. */

1 **begin**

2 $\quad \boldsymbol{q}_\lambda \leftarrow 1/(\tilde{\boldsymbol{q}}^T\tilde{\boldsymbol{q}}) \times \left(\|\tilde{\boldsymbol{q}}\|\,\tilde{\boldsymbol{q}}_\lambda - (\tilde{\boldsymbol{q}}_\lambda^T\tilde{\boldsymbol{q}}) \frac{\tilde{\boldsymbol{q}}}{\|\tilde{\boldsymbol{q}}\|} \right)$

3 $\quad$ **return** $\boldsymbol{q}_\lambda$

4 **end**

Figure 5.7: Pseudo-code description for the proposed algorithm (*cont.*)

Chapter 6

Numerical Experimentation

In this chapter, the validity of the proposed algorithm in computing both frequency- and time-domain sensitivity is demonstrated through three test cases.

6.1 Frequency-Domain Sensitivity

Figure 6.1 shows a circuit with seven distributed elements (transmission lines). These elements are identical and each has $R = 1\ \Omega/\text{cm}$, $L = 100\ \text{nH/cm}$, $G = 0.01\ \text{m}\mho/\text{cm}$, and $C = 4\ \text{pF/cm}$ as per-unit-length parameters. The length of each line is 20 cm. The distributed elements are each discretized using 70 equivalent lumped **RLGC** sections.

The proposed algorithm was then run to compute the derivative of the orthogonal basis used in the reduction. Figure 6.2 shows the frequency response of the circuit at V_{out}. The computed derivative was utilized to compute the frequency-domain sensitivity of V_{out} w.r.t. the PUL inductance of line 1 (L_1) (shown in Figure 6.3), capacitance of line 7 (C_7) (shown in Figure 6.4), resistance of line 5 (R_5) (shown in Figure 6.5), and the length of line 4 (d_4) (shown in Figure 6.6). Also shown on the graphs are comparisons with a perturbation approach used to calculate the sensitivities using a SPICE-based simulation of the circuit.

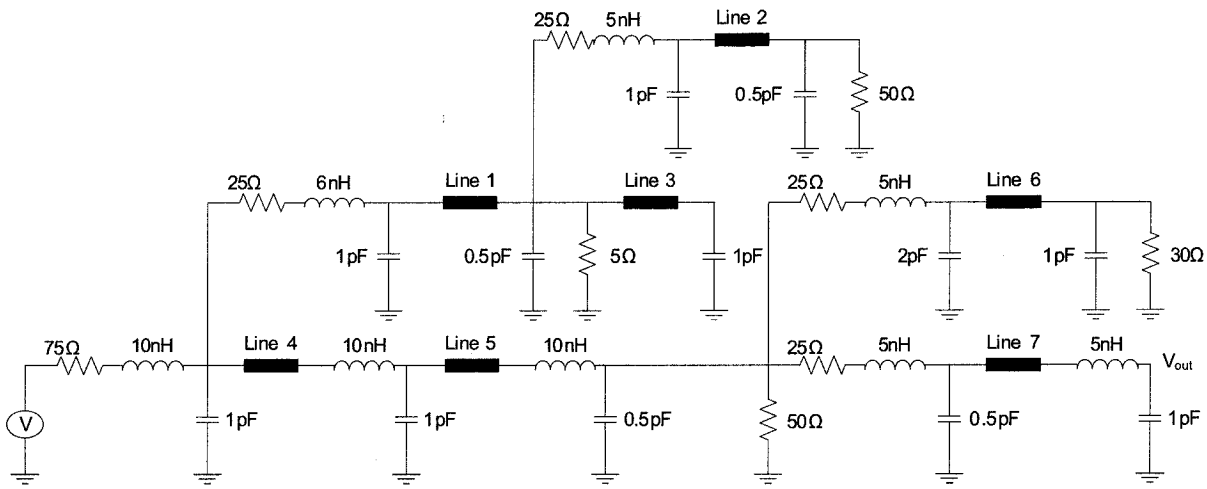


Figure 6.1: Linear circuit with 7 distributed elements (example 1)

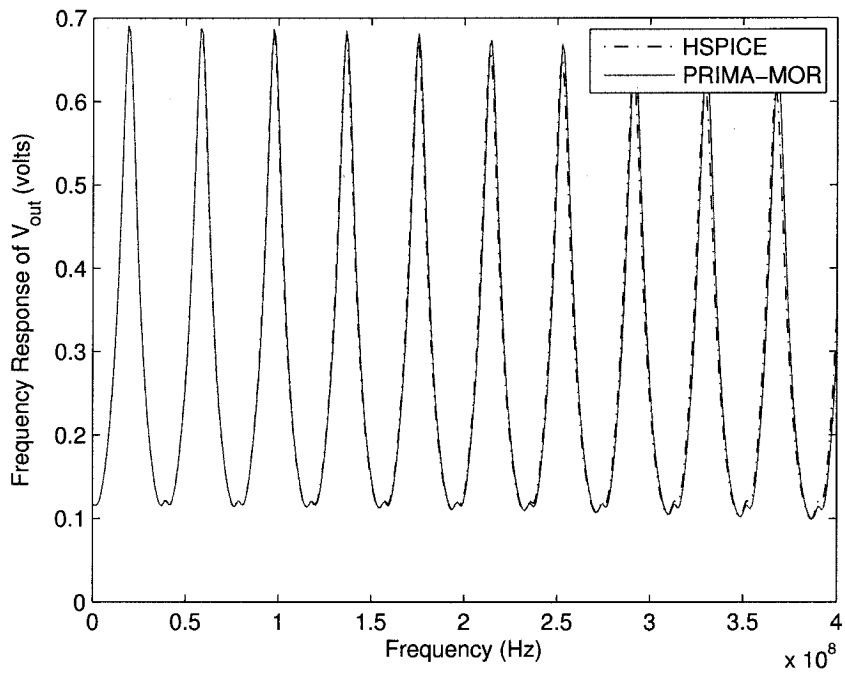


Figure 6.2: Frequency response of the circuit (Figure 6.1) at V_{out}

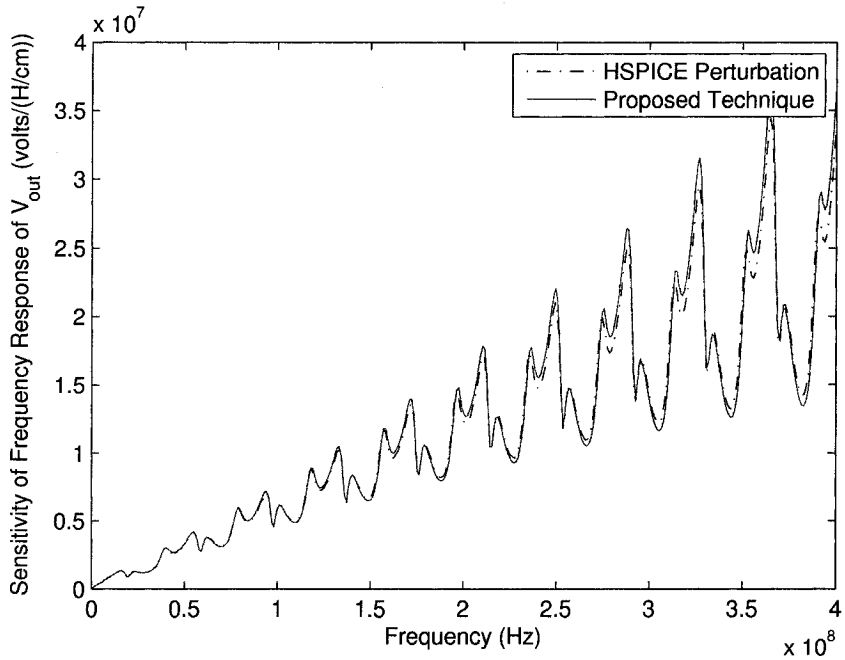


Figure 6.3: Sensitivity of the circuit response at V_{out} in the frequency domain w.r.t. L_1

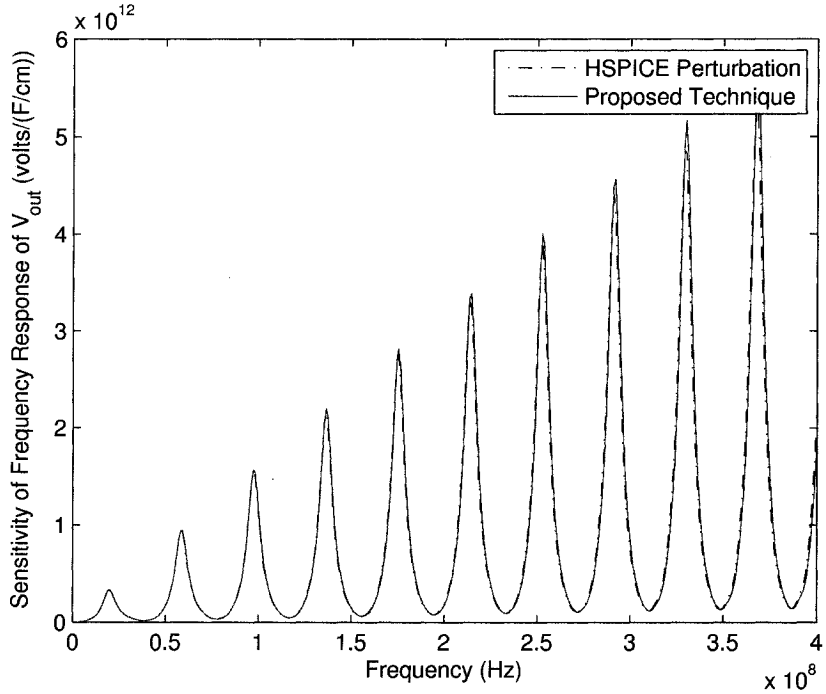


Figure 6.4: Sensitivity of the circuit response at V_{out} in the frequency domain w.r.t. C_7

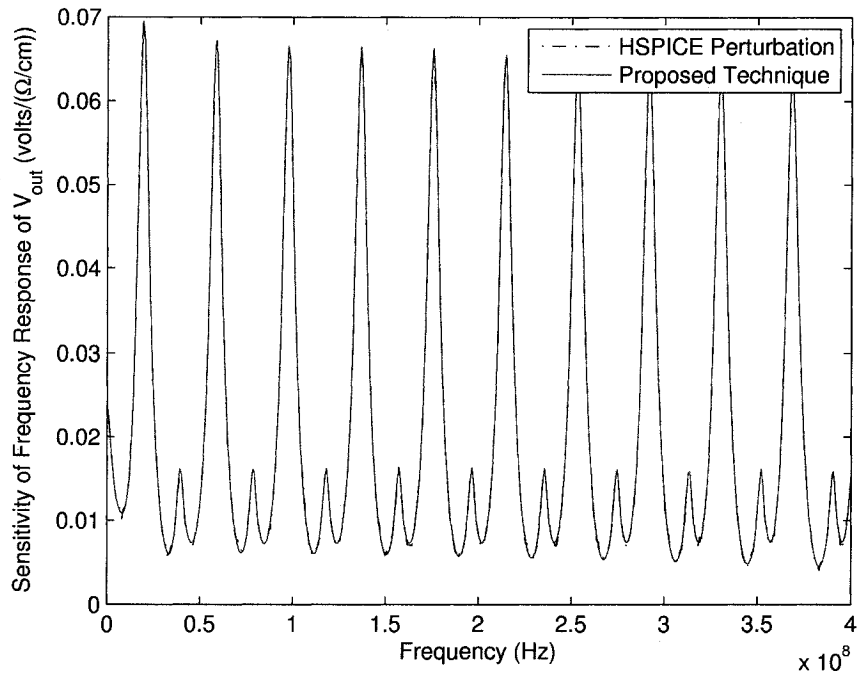


Figure 6.5: Sensitivity of the circuit response at V_{out} in the frequency domain w.r.t. R_5

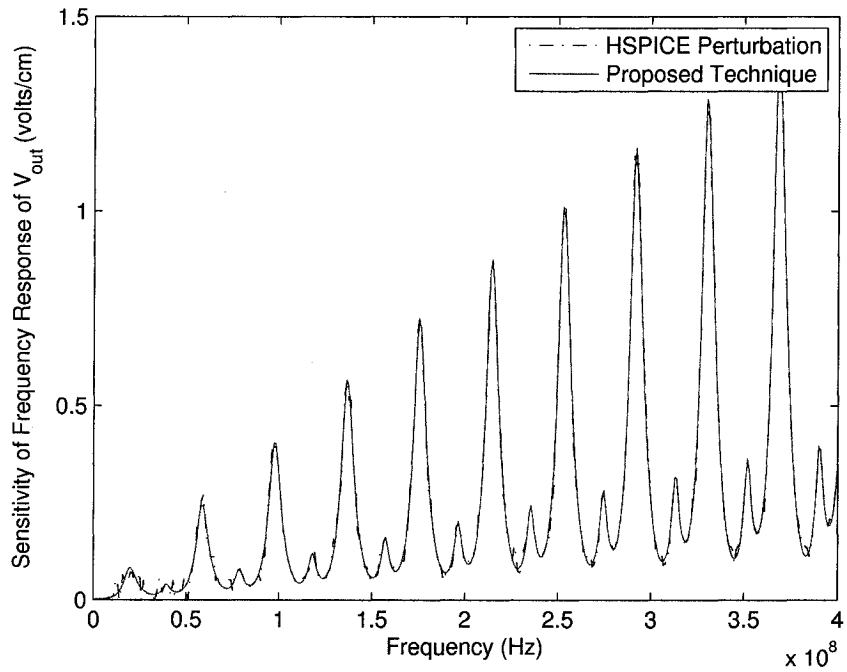


Figure 6.6: Sensitivity of the circuit response at V_{out} in the frequency domain w.r.t. d_4

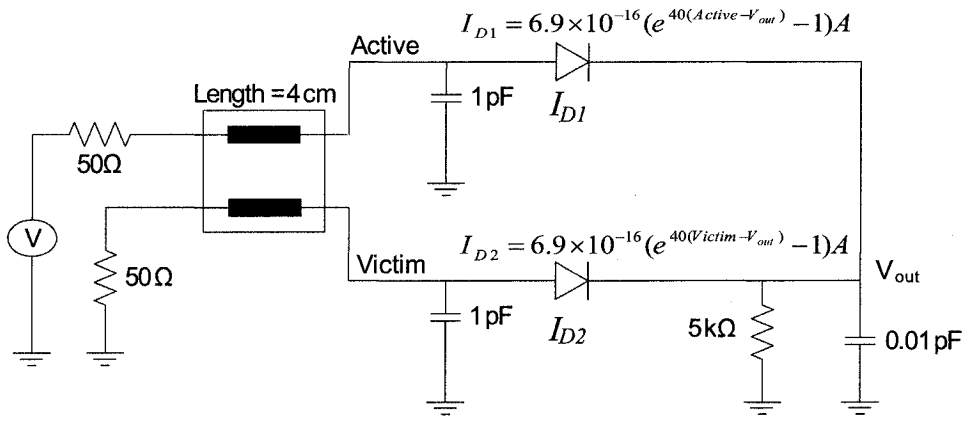


Figure 6.7: Coupled interconnect system with nonlinear terminations (example 2)

6.2 Time-Domain Sensitivity

In this example, computing the time-domain sensitivity of the nonlinear circuit shown in Figure 6.7 is considered. The length of each transmission lines is 4 cm and the PUL parameters are given as follows,

$$\mathcal{R} = \begin{bmatrix} 0.14 & 0 \\ 0 & 0.14 \end{bmatrix} \Omega/\text{cm} \qquad \mathcal{L} = \begin{bmatrix} 6.3 & 2.9 \\ 2.9 & 6.3 \end{bmatrix} \text{nH/cm}$$

$$\mathcal{C} = \begin{bmatrix} 1.1 & -0.45 \\ -0.45 & 1.1 \end{bmatrix} \text{pF/cm}$$

A lumped segmentation model was used to discretize the distributed elements and MOR technique was applied to reduce the size of the resulting two linear subnetworks down to 50×50 nodal variables for each subnetwork. The time-domain responses of the far-end voltages and V_{out} are shown in Figures 6.8, 6.9 and 6.10 in response to 5V trapezoidal excitation with rise and fall times of $0.1\ ns$ and width of $0.8\ ns$.

Next, we proceeded to compute the sensitivity of far-end active and victim voltages and V_{out} w.r.t. the parameter $C_{1,1}$ of the PUL capacitance matrix at the nominal design value of

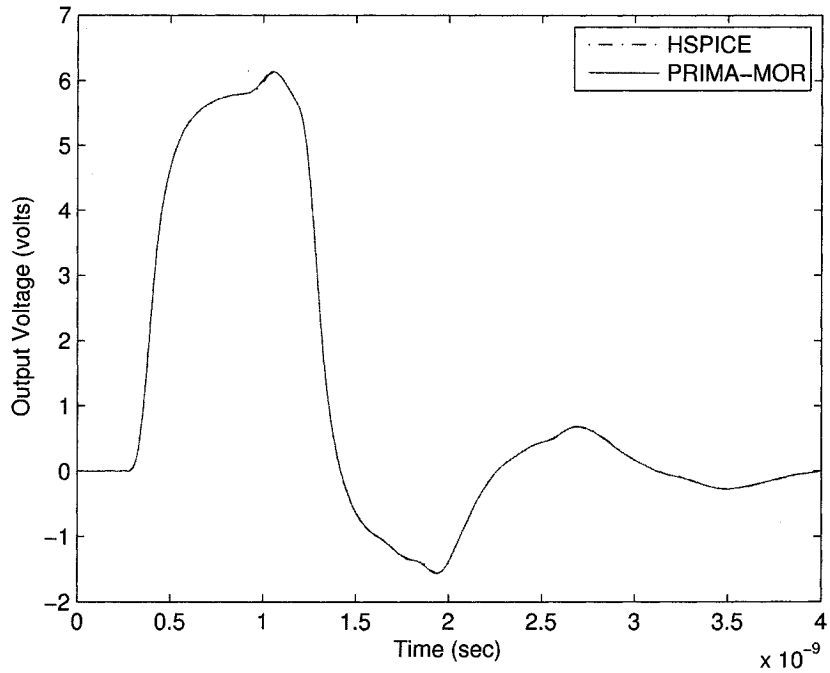


Figure 6.8: Active far-end response

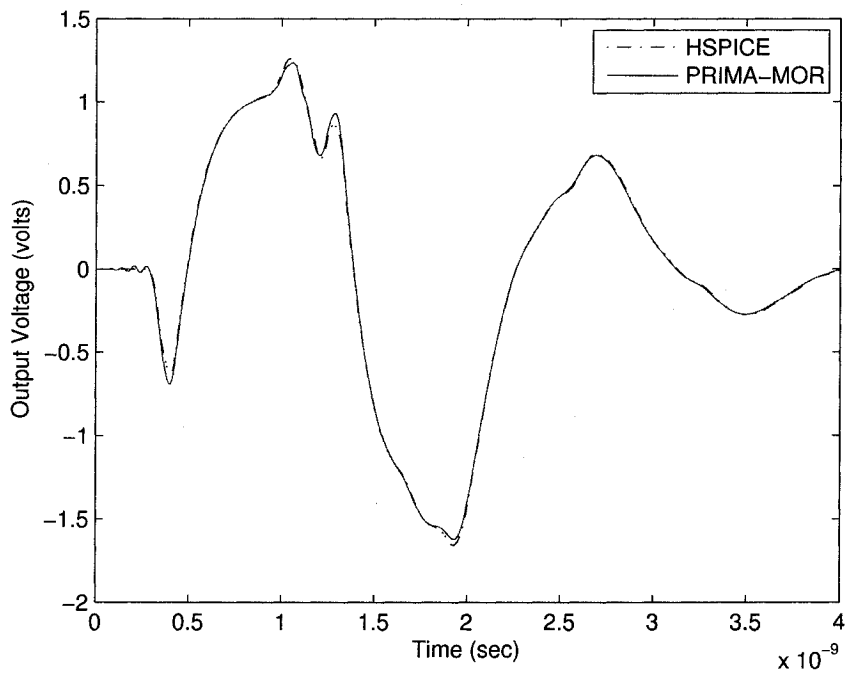


Figure 6.9: Victim far-end response

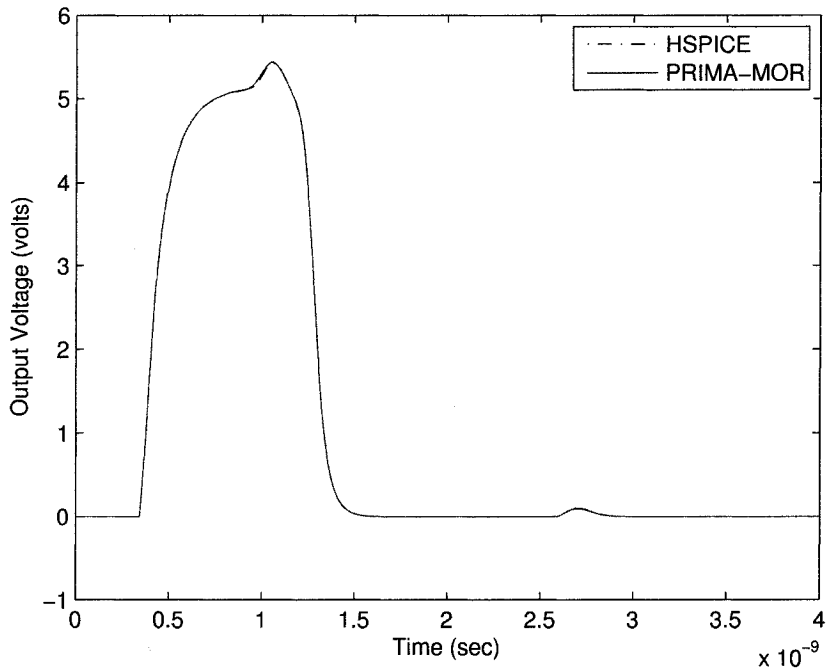


Figure 6.10: V_{out} response

1.1 pF/cm. Figures 6.12, 6.13 and 6.14 show the sensitivities obtained from the proposed algorithm and compare them with SPICE-based perturbation approach. It is to be noted that perturbation-based techniques can lead to inaccurate results (depending on the magnitude of the perturbation and the stiffness of the problem in hand). Figure 6.11 shows that the accuracy of sensitivity information obtained via perturbation-based technique is *highly* dependent on the perturbation width and the local stiffness of the response waveform. In addition, the nonlinear differential equations representing the perturbed network must be solved separately for every parameter of interest. However, in the proposed approach, the sensitivity information with respect to all the parameters can be obtained from the solution of the original and adjoint networks.

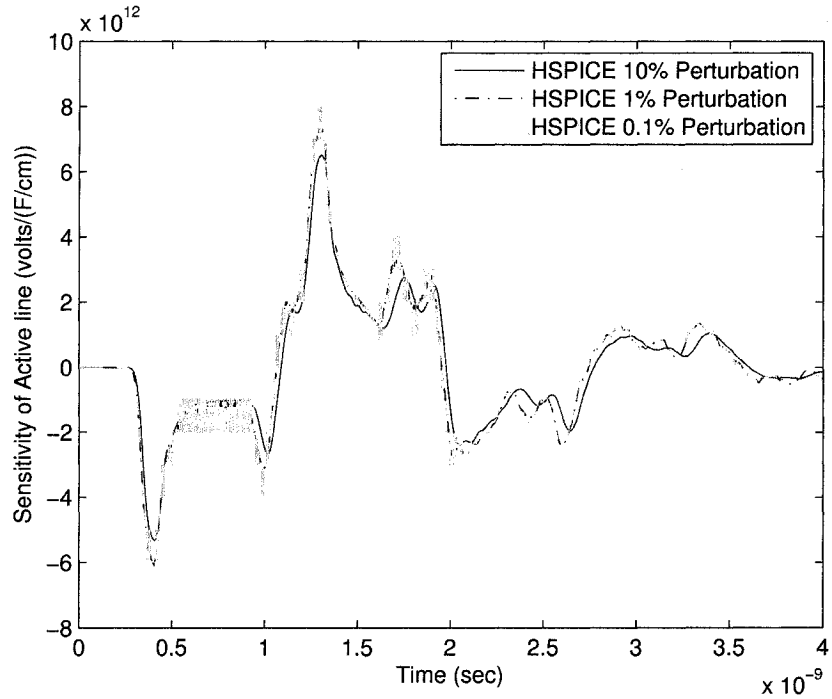


Figure 6.11: Sensitivity of active far-end using HSPICE perturbations

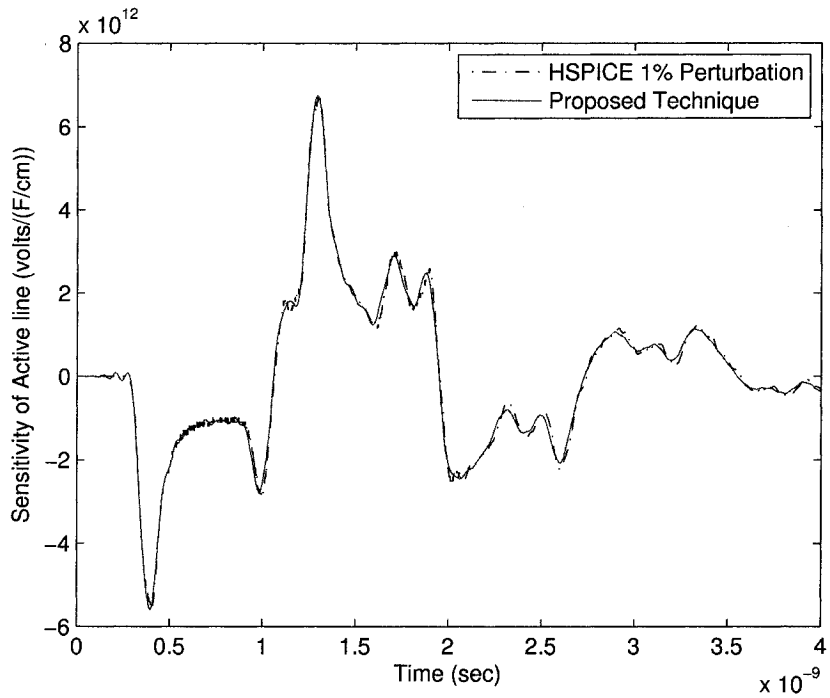


Figure 6.12: Sensitivity of active far-end with respect to $C_{1,1}$

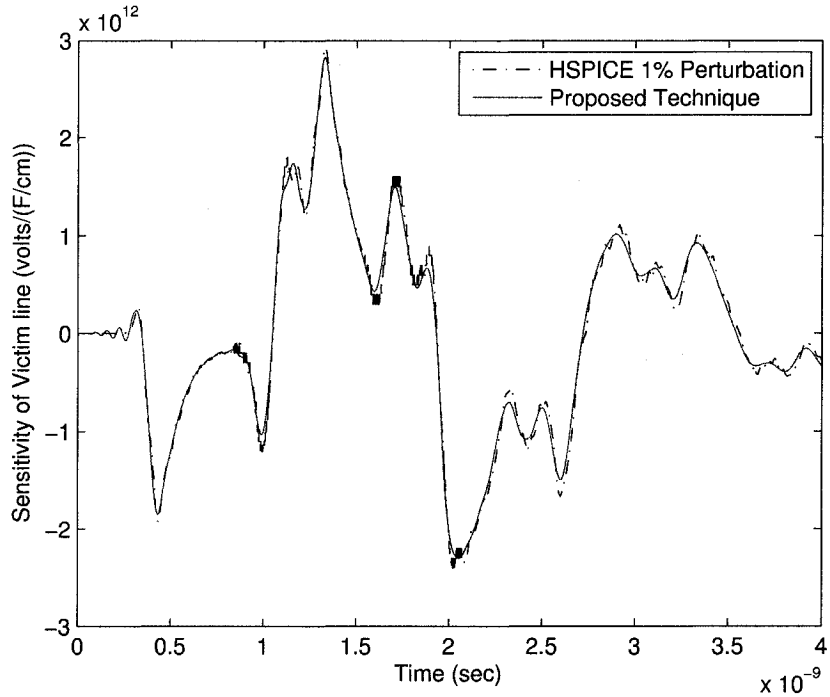


Figure 6.13: Sensitivity of victim far-end with respect to $C_{1,1}$

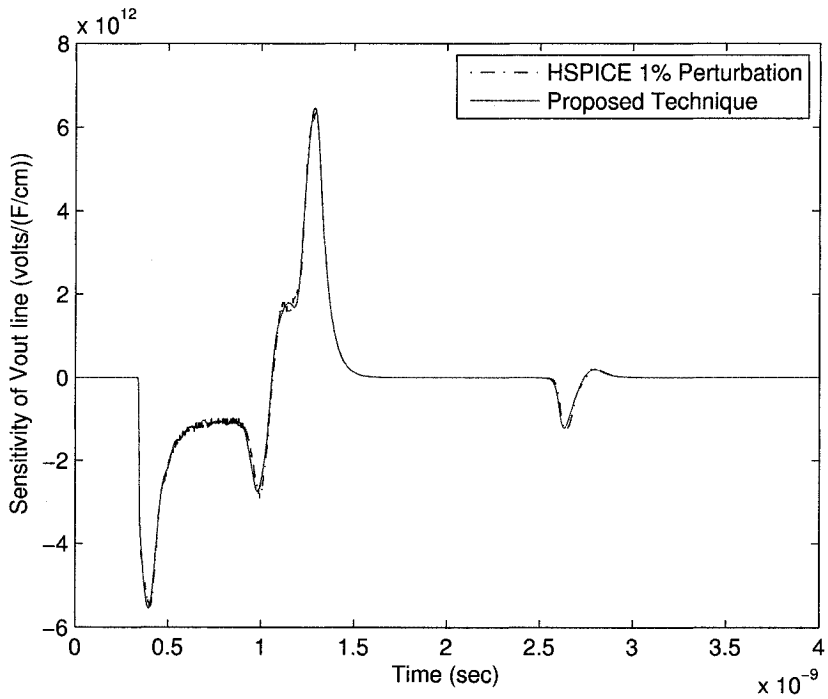


Figure 6.14: Sensitivity of V_{out} with respect to $C_{1,1}$

6.3 Optimization of Time-Domain Waveforms

In this example, time-domain sensitivity information obtained by the proposed algorithm are utilized to perform optimization on the circuit shown in Figure 6.15. Physical parameters of distributed elements (interconnects) are to be optimized to achieve certain design specifications at different nodes. The circuit shown in Figure 6.15 contains three interconnects of microstrip layout. Each interconnect consists of two coupled signal conductors. The layout of these interconnects is shown in Figure 6.16 where the interconnect conductor width w, the separation distance s, the board thickness h, and line lengths d_1, d_2 and d_3 are the interconnect parameters to be optimized. It is assumed that the three interconnects share the same value for w, s, and g. The driving excitation V is a unit step of peak 5 volts and rise time of 1 ns which drives the circuit. The nodal responses of interest are labeled V_1, V_2, V_3, and V_4 where the waveforms to be optimized. The unit step responses before and after optimization are shown in Figures 6.17-6.20 along with the design specifications. The design specification mandates that,

1. The delay time of V_1 is to be decreased to 5.5 ns based on 3 $volts$ threshold.
2. The delay time of V_3 is to be decreased to 5 ns based on 3 $volts$ threshold.
3. The response of V_1 is to be maintained greater than 4 $volts$ after 6.5 ns.
4. The response of V_3 is to be maintained greater than 4 $volts$ after 6 ns.
5. The magnitude of the responses at V_2 is to be strictly maintained less than 0.4 $volts$ all the time.
6. The magnitude of the responses at V_4 is to be strictly maintained less than 0.4 $volts$ all the time.

The error function e_i^c is the error function number i that corresponds to design specification c stated above. These error functions are given by,

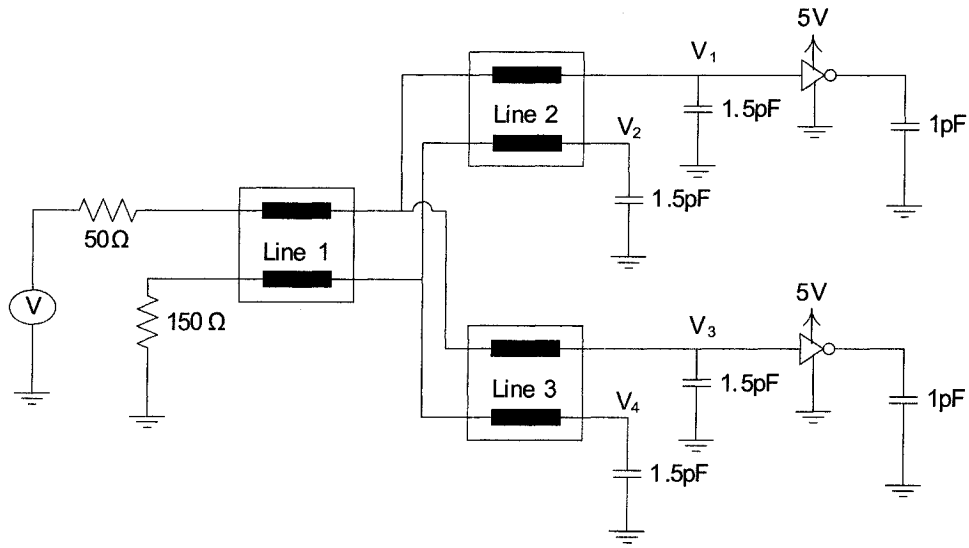


Figure 6.15: Three coupled interconnects system with nonlinear terminations (example 3)

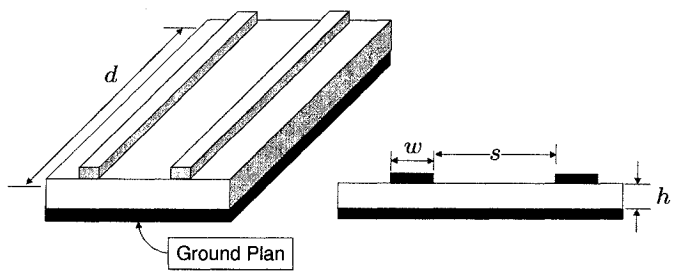


Figure 6.16: microstrip layout of two-conductor interconnects

$$
\begin{aligned}
e_1^1 &= [V_1(\phi, 5.5ns) - 3] \\
e_3^2 &= [V_3(\phi, 5ns) - 3] \\
e_{i+4}^3 &= -[V_1(\phi, t_i) - 4] \\
e_{k+36}^5 &= 10\,[V_2(\phi, t_k) - 0.4] \\
e_{k+68}^6 &= 10\,[V_4(\phi, t_k) - 0.4] \\
e_2^1 &= -[V_1(\phi, 5.5ns) - 3] \\
e_4^2 &= -[V_3(\phi, 5ns) - 3] \\
e_{j+20}^4 &= -[V_3(\phi, t_j) - 4] \\
e_{k+52}^5 &= -10\,[V_2(\phi, t_k) + 0.4] \\
e_{k+84}^6 &= -10\,[V_4(\phi, t_k) + 0.4V]
\end{aligned}
$$

where t_i, t_j and t_k, $i, j, k = 1, 2, \ldots, 16$ are linearly spaced time points between the intervals $[6.5\ ns, 40\ ns]$, $[6\ ns, 40\ ns]$, and $[0\ ns, 40\ ns]$, respectively. The vector of designable variables is thus given by $\phi = [w\ s\ h\ d_1\ d_2\ d_3]^T$. Additional constraints imposed on the design require that total wire lengths should be $1.35\ m$, the total of signal conductor widths and the separation between them is $2.5\ mm$. The minimum width allowed is $0.1\ mm$ and h should range between $0.5\ mm$ and $2.5\ mm$. These constraints could be mathematically expressed as $g_1(\phi) = d_1 + d_2 + d_3 - 1.35\ m$, $g_2(\phi) = 2w + s - 2.5\ mm$, $w \geq 0.1\ mm$, and $0.5\ mm \leq h \leq 2.5\ mm$ The initial value for ϕ is $[0.5 \times 10^{-3}\ 1.5 \times 10^{-3}\ 2 \times 10^{-3}\ 0.45\ 0.45\ 0.45]^T\ (m)$. It is seen from Figures 6.17-6.20 that all of the design specifications were not met before optimization. The per-unit-length parameters are computed using *LINPAR* line modeling tool [40]. A grid of per-unit-length parameters is generated and the intermediate values are interpolated using surface spline. This results in more accurate values for the line electrical parameters

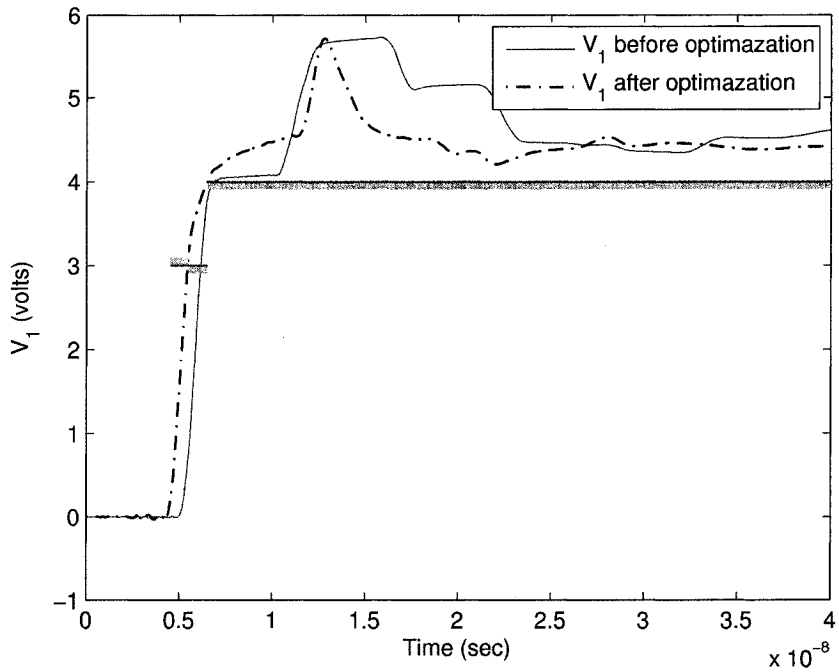


Figure 6.17: V_1 response

without invoking the modeling tool at every point of interest.

The distributed elements are discretized using equivalent 70 **RLGC** segments. An equivalent model is obtained such that it preserves the first 50 dominant poles of every discretized subnetwork. The optimization is conducted on the reduced size network. The optimization problem is linearized at every optimization step using the proposed algorithm. This enables us to use a comprehensive optimization technique that makes use of the availability of information about the first derivative. The value of ϕ after optimization is:

$$\phi = \left[0.1 \times 10^{-3}\ 2.3 \times 10^{-3}\ 0.75 \times 10^{-3}\ 0.0981\ 0.6651\ 0.5868\right]^T\ (m)$$

It could be seen from Figures 6.17-6.20 that the response after optimization is strictly met for every design specification.

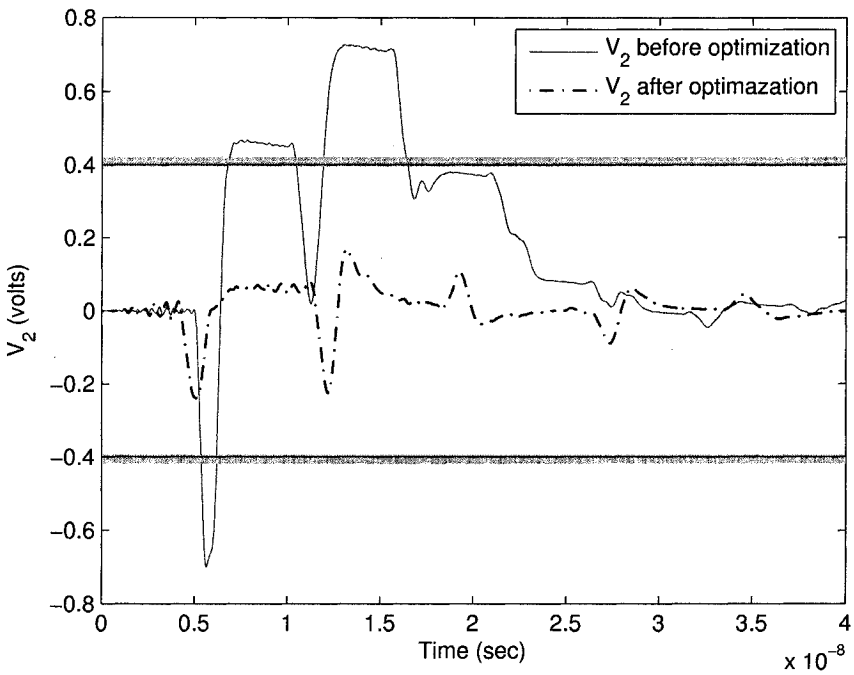


Figure 6.18: V_2 response

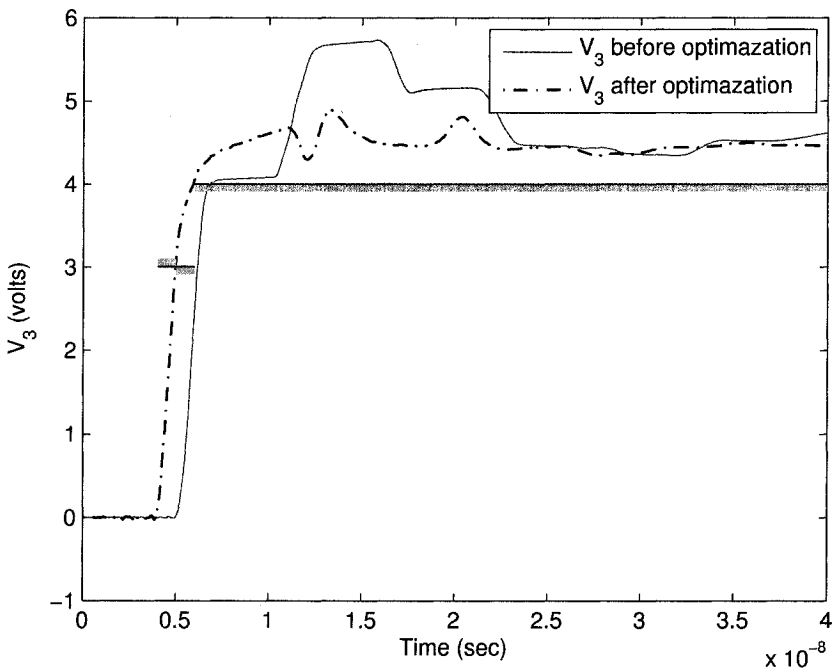


Figure 6.19: V_3 response

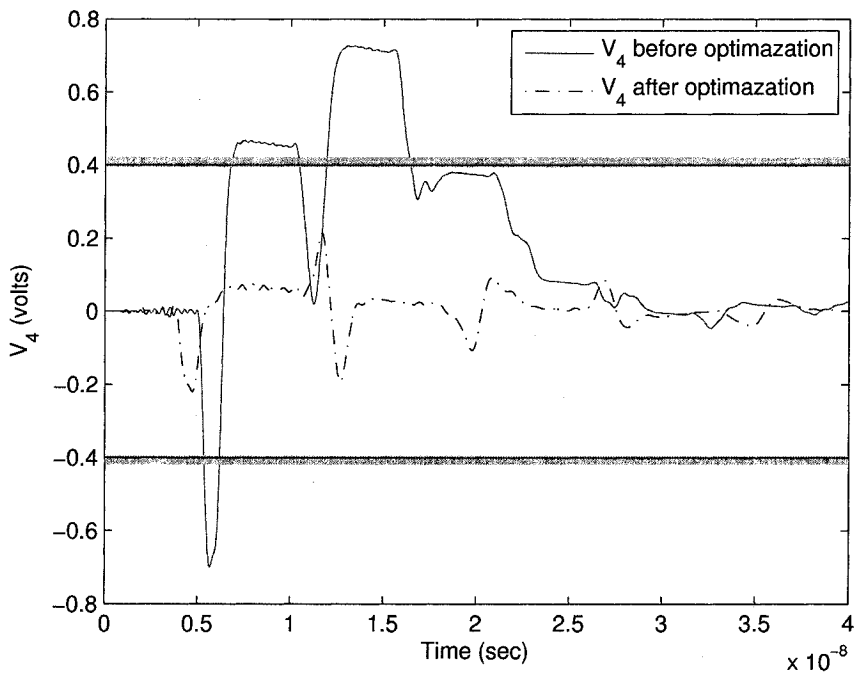


Figure 6.20: V_4 response

Chapter 7

Conclusions

This thesis presented a natural extension to Model-Order Reduction techniques by providing and implementing an efficient algorithm to obtain information about the first derivative. With the aid of this information, adjoint-based sensitivity techniques were applied to reduced models. The integration of MOR and adjoint-based sensitivity technique has the potential of enabling direct optimization of reduced models. This integration will shrink the time window required to design and fine tune products. This time reduction comes along with sufficient accuracy required by nowadays designs.

The accuracy of the proposed algorithm has been shown with a typical design problem of optimizing high speed interconnect parameters to meet strict design specifications. The high speed interconnects were discretized into a number of **RLGC** segments. The resulting large circuit was reduced in size using PRIMA [1] algorithm. The resulting values from the optimization process accurately met the design specification for the original circuit.

The proposed techniques has also the potential of carrying out optimization in the time domain, a natural domain for describing nonlinearities of electrical networks.

Also, the proposed algorithm is not limited to time-domain design problems but could be

[1] a typical model order reduction technique that preserves passivity

used in the frequency domain. Again, the accuracy of the proposed algorithm was shown for a complex circuit with seven high speed interconnects. The results were shown to be more accurate than those produced with HSPICE perturbations.

7.1 Future Work

This work could be extended in various aspects. The second derivative could be devised in a similar fashion to the technique presented in this thesis. This could enable the use of more sophisticated optimization methods.

Appendix A

Proof of Equivalence

IMM $\sim$ EMM

Moments of the reduced system are, in fact, the same as of the original system as long as the projection matrices are nonsingular and in the space of *Krylov* subspace(s). Consider the first moment ($\mathcal{M}_0$) of the reduced system:

$$\begin{aligned}\hat{\mathcal{M}}_0 &= \hat{\mathcal{L}}^T\hat{\mathcal{R}} = \hat{\mathcal{L}}^T\hat{\mathcal{G}}^{-1}\hat{\mathcal{B}} = \mathcal{L}^T\mathcal{Q}(\mathcal{P}^T\mathcal{G}\mathcal{Q})^{-1}\mathcal{P}^T\mathcal{B} \\ &= \mathcal{L}^T\mathcal{Q}(P^T\mathcal{G}\mathcal{Q})^{-1}(\mathcal{P}^T\mathcal{G}\mathcal{Q})v_0 = \mathcal{L}^T\mathcal{Q}v_0 = \mathcal{L}^T\mathcal{G}^{-1}\mathcal{B} = \mathcal{M}_0 \qquad \text{(A.1)}\end{aligned}$$

where $\mathcal{V}_0$ is an arbitrary set of vectors such that $\mathcal{G}^{-1}\mathcal{B} = \mathcal{Q}\mathcal{V}_0$[1].

Now, we proceed to compute second moment of the reduced system $\hat{\mathcal{M}}_1$:

[1] since $\mathcal{G}^{-1}\mathcal{B}$ is in column space of $\mathcal{Q}$

$$\begin{aligned}
\hat{\mathcal{M}}_1 &= \hat{\mathcal{L}}^T \hat{\mathcal{A}} \hat{\mathcal{R}} = \mathcal{L}^T \mathcal{Q} (\mathcal{P}^T \mathcal{G} \mathcal{Q})^{-1} (\mathcal{P}^T \mathcal{C} \mathcal{Q}) (\mathcal{P}^T \mathcal{G} \mathcal{Q})^{-1} \mathcal{P}^T \mathcal{B} \\
&= \mathcal{L}^T \mathcal{Q} (\mathcal{P}^T \mathcal{G} \mathcal{Q})^{-1} \mathcal{P}^T \mathcal{C} \mathcal{Q} \mathcal{V}_0 \\
&= \mathcal{L}^T \mathcal{Q} (\mathcal{P}^T \mathcal{G} \mathcal{Q})^{-1} \mathcal{P}^T \mathcal{G} \underbrace{\mathcal{G}^{-1} \mathcal{C} \mathcal{G}^{-1} \mathcal{B}}_{\mathcal{Q} \mathcal{V}_1} \\
&= \mathcal{L}^T \mathcal{Q} (\mathcal{P}^T \mathcal{G} \mathcal{Q})^{-1} \mathcal{P}^T \mathcal{G} \mathcal{Q} \mathcal{V}_1 \\
&= \mathcal{L}^T \mathcal{Q} v_1 = \mathcal{L}^T \mathcal{A} \mathcal{R} = \mathcal{M}_1
\end{aligned} \tag{A.2}$$

where the i^{th} $(0 \leq i \leq \lceil \frac{q}{N} \rceil - 1)$ sub-block matrices of Krylov subspace could be expressed as:

$$(\mathcal{G}^{-1} \mathcal{C})^i (\mathcal{G}^{-1} \mathcal{B}) = \mathcal{Q} \mathcal{V}_i \tag{A.3}$$

Proof that subsequent moments match should follow systematically as shown in Equation (A.1) and Equation (A.2). And hence, it could be shown that the reduced system is equivalent to the original system up to a limited number of moments q. If matrix $\mathcal{P}$ is defined as in (2.17), the reduced system will match $2q$ moments[2]. We prove that the q^{th} moment of the reduced systems matches q^{th} of the original system.

We know that $\mathcal{G}^{-T} \mathcal{L}$ could be expressed as:

$$\mathcal{G}^{-T} \mathcal{L} = \mathcal{P} \mathcal{W}_0 \tag{A.4}$$

[2]Assuming that $2q$ is less than the number of controllable states plus the number of observable states.

$$
\begin{aligned}
\hat{\mathcal{M}}_q &= \hat{\mathcal{L}}^T \hat{\mathcal{A}}^q \hat{\mathcal{R}} = \mathcal{L}^T \mathcal{Q}\{(\mathcal{P}^T \mathcal{G}\mathcal{Q})^{-1}(\mathcal{P}^T \mathcal{C}\mathcal{Q})\}^q (\mathcal{P}^T \mathcal{G}\mathcal{Q})^{-1}\mathcal{P}^T \mathcal{B} \\
&= \mathcal{L}^T \mathcal{Q}(\mathcal{P}^T \mathcal{G}\mathcal{Q})^{-1}\mathcal{P}^T \mathcal{C} \underbrace{\mathcal{Q}\{(\mathcal{P}^T \mathcal{G}\mathcal{Q})^{-1}(\mathcal{P}^T \mathcal{C}\mathcal{Q})\}^{q-1}(\mathcal{P}^T \mathcal{G}\mathcal{Q})^{-1}\mathcal{P}^T \mathcal{B}}_{\mathcal{A}^{q-1}\mathcal{R}} \\
&= \underbrace{\mathcal{L}^T \mathcal{G}^{-1}}_{\mathcal{W}_0^T \mathcal{P}^T} \mathcal{G}\mathcal{Q}(\mathcal{P}^T \mathcal{G}\mathcal{Q})^{-1}\mathcal{P}^T \mathcal{C}\mathcal{A}^{q-1}\mathcal{R} \\
&= \mathcal{W}_0^T \mathcal{P}^T \mathcal{G}\mathcal{Q}(\mathcal{P}^T \mathcal{G}\mathcal{Q})^{-1}\mathcal{P}^T \mathcal{C}\mathcal{A}^{q-1}\mathcal{R} \\
&= \underbrace{\mathcal{W}_0^T \mathcal{P}^T}_{\mathcal{L}^T \mathcal{G}^{-1}} \mathcal{C}\mathcal{A}^{q-1}\mathcal{R} \\
&= \mathcal{L}^T \mathcal{G}^{-1}\mathcal{C}\mathcal{A}^{q-1}\mathcal{R} = \mathcal{L}^T \mathcal{A}^q \mathcal{R} = \mathcal{M}_q
\end{aligned}
\tag{A.5}
$$

Appendix B

Proof of Equivalence

MGS $\sim$ CGS

To show the equivalence between the two update procedures, we note that by the orthonormality of two matrices $\mathcal{Q}_u$ and $\mathcal{Q}_v$, we have $\mathcal{Q}_u^T \mathcal{Q}_v = \mathbf{0}$ for $(u \neq v)$ and $\mathcal{Q}_u^T \mathcal{Q}_u = \mathcal{I}$ and, therefore,

$$\frac{\partial \mathcal{Q}_u}{\partial \lambda_{i,j}}^T \mathcal{Q}_v + \mathcal{Q}_u^T \frac{\partial \mathcal{Q}_v}{\partial \lambda_{i,j}} = \mathbf{0}. \tag{B.1}$$

At the beginning of k^{th} step of the modified update procedure in Figure 5.4, it can be seen that both $\tilde{\mathcal{Q}}_{\lambda,m}$ and $\tilde{\mathcal{Q}}_m$ are given in terms of the following linear combination

$$\begin{aligned} \tilde{\mathcal{Q}}_{\lambda,m} &= \frac{\partial \xi_{m-1}}{\partial \lambda_{i,j}} + \sum_{r=0}^{k-1} \mathcal{Q}_r \mathcal{E}_r + \frac{\partial \mathcal{Q}_r}{\partial \lambda_{i,j}} \mathcal{P}_r \\ \tilde{\mathcal{Q}}_m &= \xi_{m-1} + \sum_{r=0}^{k-1} \mathcal{Q}_r \mathcal{P}_r \end{aligned} \tag{B.2}$$

where $\mathcal{E}_r$ and $\mathcal{P}_r$ are matrices of size $P_{\pi_i} \times P_{\pi_i}$. On the k^{th} step, both of $\tilde{\mathcal{Q}}_{\lambda,m}$ and $\tilde{\mathcal{Q}}_m$ are operated on in the following inner-products, $\left\langle \tilde{\mathcal{Q}}_{\lambda,m}, \mathcal{Q}_k \right\rangle$, $\left\langle \tilde{\mathcal{Q}}_m, \frac{\partial \mathcal{Q}_k}{\partial \lambda_{i,j}} \right\rangle$ and $\left\langle \tilde{\mathcal{Q}}_m, \mathcal{Q}_k \right\rangle$. Using Equation (B.1) and the orthonormality property between $\mathcal{Q}_k$ with $\mathcal{Q}_r$ $(r = 0, \cdots, k-1)$,

these inner-products reduce to the following expressions:

$$\begin{aligned}
\left\langle \tilde{\mathcal{Q}}_{\lambda,m}, \mathcal{Q}_k \right\rangle &= \left\langle \frac{\partial \boldsymbol{\xi}_{m-1}}{\partial \lambda_{i,j}}, \mathcal{Q}_k \right\rangle + \sum_{r=0}^{k-1} \left\langle \frac{\partial \mathcal{Q}_r}{\partial \lambda_{i,j}}, \mathcal{Q}_k \right\rangle \mathcal{P}_r \\
\left\langle \tilde{\mathcal{Q}}_m, \frac{\partial \mathcal{Q}_k}{\partial \lambda_{i,i}} \right\rangle &= \left\langle \boldsymbol{\xi}_{m-1}, \frac{\partial \mathcal{Q}_k}{\partial \lambda_{i,j}} \right\rangle + \sum_{r=0}^{k-1} \left\langle \mathcal{Q}_r, \frac{\partial \mathcal{Q}_k}{\lambda_{i,j}} \right\rangle \mathcal{P}_r \\
\left\langle \tilde{\mathcal{Q}}_m, \mathcal{Q}_k \right\rangle &= \langle \boldsymbol{\xi}_{m-1}, \mathcal{Q}_k \rangle
\end{aligned} \tag{B.3}$$

From the above expressions for the inner-products and using Equation (B.1), it can be seen that the k^{th} update of Figure 5.4 reduces to the right-side of (5.16).

Appendix C

Glossary of Terms

CAD	Computer Aided Design
CFH	Complex Frequency Hopping
CGS	Classical Gram Schmidt CPU Central Processing Unit
DAE	Deferential Algebraic Equations
DFT	Direct Fourier Transform
EDA	Electronic Design Automation
EMM	Explicit Moment Matching
F/B	Forward Backward
FDTD	Finite Difference Time Domain
FEM	Finite Element Method
FFT	Fast Fourier Transform
IMM	Implicit Moment Matching
L/U	Lower Upper
LTI	Linear Time Invariant
MGS	Modified Gram Schmidt
MIMO	Multi-Input Multi-Output

MNA	Modified Nodal Analysis
MOR	Model Order Reduction
MTL	Multi-conductor Transmission Line
MVP	Matrix-Vector Product
NR	Newron Raphson method
PEEC	Partial Element Equivalent Circuit
PRIMA	Passive Reduced-Order Interconnect Macromodeling Algorithm
PUL	per unit length
RC	Resistor-Capacitor Network
RF	Radio Frequency
RLC	Resistor-Inductor-Capacitor
SISO	Single-Input Single-Output
SoC	System on Chip
TE	Truncation Error
VLSI	Very Large Scale Integration

Bibliography

[1] R. Achar and M. Nakhla. Simulation of high-speed interconnects. *Proceedings IEEE*, 89(5):693–728, May 2001.

[2] Wendemagegnehu T. Beyene and José E. Schutt-Ainé. Krylov subspace-based model-order reduction techniques for circuits simulations. pages 331–334, 1997.

[3] B. Denecker, F. Olyslager, L. Knockhaert, and D. D. Zutter. Generation of FDTD sub-cell equations by means of reduced order modeling. *IEEE Trans. Antennas Propagat.*, 51(8):1806–1817, August 2003.

[4] L. Kulas and M. Mrozowski. Reduced order models of refined yee's cells. *IEEE Microwave and Wireless Components Letters*, 13(4):164–166, April 2003.

[5] J. Vlach and K. Singhal. *Computer Methods for Circuit Analysis and Design*. Van Norstrand Reinolds, New York, 1983.

[6] T. Ahmed, E. Gad, and M. C. E. Yagoub. An adjoint-based approach to computing the time-domain sensitivity of systems described by reduced-order model. In *IEEE MTT-S Int. Microwave Symp. Dig.*, pages 1783–1786, June 2005. Long Beach, California, U.S.A.

[7] T. Ahmed, E. Gad, and M. C. E. Yagoub. An adjoint-based approach to computing time-domain sensitivity of multiport systems described by reduced-order models. *IEEE Trans. Microwave Theory Tech.*, 53(11):3538–3547, November 2005.

[8] L. Kulas and M. Mrozowski. Reduced-order models in FDTD. *IEEE Microwave and Wireless Components Letters*, 11(10):422–424, October 2001.

[9] L. Kulas and M. Mrozowski. A fast high-resolution 3-d finite-differerence time-domain scheme with macromodels. *IEEE Trans. Microwave Theory Tech.*, 52(9):2330–2335, September 2004.

[10] J. Dongarra and F. Sullivan. Guest editors' introduction: The top 10 algorithms. 2(1):22–23, January 2000.

[11] R. Achar, P. Gunupudi, M. Nakhla, and E. Chiprout. Passive interconnect reduction algorithm for distributed/measured networks. *IEEE Trans. Circuits Syst. II*, 47(4):287–301, April 2000.

[12] A. Dounavis, E. Gad, and M. Nakhla. Passive model-reduction of multiport distributed networks including frequency dependent parameters. *IEEE Trans. Microwave Theory Tech.*, 48(12):2325–2334, December 2000.

[13] A Odabasioglu, M. Celik, and L. T. Pileggi. PRIMA: passive reduced-order interconnect macromodeling algorithm. *IEEE Trans. Computer-Aided Design*, 17(8):645–654, August 1998.

[14] S. Pasha, A. C. Cangellaris, and J. Prince. An all-purpose dispersive multiconductor interconnect model compatible with PRIMA. *IEEE Trans. Adv. Packag.*, 24(2):126–131, May 2001.

[15] A. Odabasioglu, M. Celik, and L.T. Pileggi. Practical considerations for passive reduction of RLC circuits. pages 214–219, 1999. San Jose, California, U.S.A.

[16] S. Director. *Computer-aided circuit design: simulation and optimization.* Hutchinson & Ross, 1974.

[17] R Khazaka, P. Gunupudi, and M. S. Nakhla. Efficient sensitivity analysis of transmission-line networks using model reduction techniques. *IEEE Trans. Microwave Theory Tech.*, 48(12):2345–2351, December 2000.

[18] S. Lum, M. S. Nakhla, and Q. J. Zhang. Sensitivity analysis of lossy coupled transmission lines with nonlinear terminations. *IEEE Trans. Microwave Theory Tech.*, 42(4):607–615, April 1994.

[19] S. Lum, M. S. Nakhla, and Q. J. Zhang. Sensitivity analysis of lossy coupled transmission lines. *IEEE Trans. Microwave Theory Tech.*, 39(12):2089–2099, December 1991.

[20] L. Daldoss, P. Gubian, and M. Quarantelli. Multiparameter time-domain sensitivity computation. *IEEE Trans. Circuits Syst. II*, 48(11):1296–1307, November 2001.

[21] S. M. Ali, N. K. Nikolova, and M. H. Bakr. Sensitivity analysis with full-wave electromagnetic solvers based on structured grids. *IEEE Trans. on Magnetics*, 40(3):1521–1529, May 2004.

[22] S. M. Ali, N. K. Nikolova, and M. H. Bakr. Central adjoint variable method for sensitivity analysis with structured grid electromagnetic solvers. *IEEE Trans. on Magnetics*, 40(4):1969–1971, July 2004.

[23] M. H. Bakr and N. Nikolova. An adjoint variable method for time-domain transmission line modeling with fixed structured grids. *IEEE Trans. Microwave Theory Tech.*, 52(2):554–559, February 2004.

[24] S. W. Director and R. A. Roher. The generalized adjoint network and network sensitivities. 16(3):318–323, August 1969.

[25] A. Dounavis, R. Achar, and M Nakhla. Efficient sensitivity analysis of lossy multiconductor transmission lines with nonlinear terminations. *IEEE MTT-S Int. Microwave Symp. Dig.*, 49(12):2292–2299, December 2001.

[26] Y. Saad. *Iterative Methods for Sparse Linear Systems.* PWD Publishing Company, Boston, 1996.

[27] B. Lohmann and B. Salimbahrami. *Methods and Applications in Automation*, chapter Introduction to Krylov Subspace Methods in Model Order Reduction, pages 1–13. Shaker Verlag, Aachen, 2003.

[28] M. Nakhla and R. Achar. *High-speed circuit and interconnect analysis.* Multimedia learning course, ON: Omniz Global Knowledge Corporation (www.omniz.com), 2001.

[29] E. Chiprout and N. Nakhla. Analysis of interconnect networks using complex frequency hopping (CFH). *IEEE Trans. Computer-Aided Design*, 14(2):186–200, February 1995.

[30] E. Chiprout and M. Nakhla. *Asymptotic Waveform Evaluation and Moment Matching for Interconnect Analysis.* Kluwer, Boston, 1993.

[31] D. H. Xie and M. Nakhla. Delay and crosstalk simulation of high-speed VLSI interconnects with nonlinear terminations. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 12(11):1798–1811, November 1993.

[32] S.Y Kim, N. Gopal, and L. T. Pillage. Time-domain macromodels for VLSI interconnect analysis. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 13(10):1257–1270, October 1994.

[33] P. Feldmann and R. W. Freund. Efficient linear cut analysis by Padé via Lanczos process. *IEEE Trans. Computer-Aided Design*, 14(5):639–649, May 1995.

[34] W. E. Arnoldi. The principle of minimized iteration in the solution of the matrix eigenproblem. *Quarterly Applied Mathematics*, 9:17–29, 1951.

[35] C. Lanczos. Solution of systems of linear equations by minimized iterations. *J. Research Nat'l Bureau of Standards*, 49:33–53, 1952.

[36] R. W. Freund and P. Feldmann. Reduced-order modeling of large linear passive multiterminal circuits using matrix-Padé approximation. In *IEEE Proceedings on Design, Automation and Test*, pages 530–537, February 1998. Le Palais des Congrès de Paris, Paris, France.

[37] T. V. Nguyen and J. Li. Multipoint Padé approximation using a rational block Lanczos algorithm. pages 72–75, 1997. San Jose, California, U.S.A.

[38] R. W. Freund. Passive reduced-order modeling via Krylov-subspace methods. In *IEEE International Symposium on Computer-Aided Control System Design, 2000. CACSD 2000.*, pages 261–266, September 2000. Anchorage, Alaska, U.S.A.

[39] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins Press, 1989.

[40] A. R. Djordjevic, M. B. Bazdar, T. K. Sarkar, and R. F. Harrington. LINPAR for Windows: Matrix parameters for multiconductor transmission lines. Software and User's Manual, Version 1.2, 1995. Artech House Inc., Norwood, MA.

[41] K. S. Kundert. Introduction to RF simulation and its application. *IEEE Journal of Solid-State Circuits*, 34(9):1298–1319, September 1999.

[42] G. H. Golub and C. Van Loan. *Matrix Computations*. John Hopkins Univ. Press, Baltimore, MD, 1989.