7-27-2020 11:15 AM

Analytic Delay Model of RLC Interconnects using Numerical Inversion of the Laplace Transform

Yu Jiang, *The University of Western Ontario*

A thesis submitted in partial fulfillment of the requirements for the Master of Engineering Science degree in Electrical and Computer Engineering

Abstract

Signal integrity analysis for on-chip interconnect becomes increasingly important in high-speed designs. SPICE, a conventional circuit simulator, can provide accurate prediction for interconnects, however, using SPICE is extremely computationally expensive. On the other hand, explicit moment matching technique can produce unstable poles for highly accurate approximations and implicit moment matching technique can obtain more accurate approximations at the expense of computational complexity. This thesis presents an analytic model to efficiently estimate the signal delays of *RLC* on-chip interconnects. It uses the numerical inversion of Laplace transform (NILT) to obtain time function, suitable for transient analysis. Since the integration formula of the NILT is numerically stable for higher order approximations, the developed algorithm provides a mechanism to increase the accuracy for delay estimation. Numerical examples are implemented and compared with HSPICE, two-pole model and Passive Reduced-Order Interconnect Macromodeling Algorithm (PRIMA) to illustrate the efficiency and validity of the proposed work.

Keywords

Interconnects, clock tree synthesis, moment matching, resistance-inductance-capacitance (*RLC*), Numerical Inversion of the Laplace Transform, 50% delay

Summary for Lay Audience

Very-large-scale-integration (VLSI) and integrated circuits (IC) are widely used in such electronic fields such as mobile, satellite communication, computer hardware, microelectromechanical systems, robotics. As the rapid decrease in feature size and significant increase in circuit complexity, density and operating speeds, applying an accurate and efficient method for analyzing on-chip interconnects becomes very important for circuit designers. The evaluated results generated by analysis methods can provide the exact descriptions for the ability or availability of a high-speed system being designed to function under stated conditions for a specified time period. Due to the improper design, the interconnect effects such as the signal delay, crosstalk, and ringing can severely degrade signal integrity (SI), i.e. a set of measures of the quality of an electrical signal, and cause false actions of the circuits, which brings about costly redesigns. As a result, designers must consider the effects of interconnects at the early stages of the design cycle.

Researchers have presented some methods, such as the Computer Aided Design (CAD) tools and Simulation Program with Integrated Circuit Emphasis (SPICE) to analyze on-chip interconnect. However, these simulation tools need a rather long time to obtain the estimated results and present higher requirements for data storage, which limits their application in practice. To solve the problems existing in simulation tools, researchers presented some analytic formulas, such as the Elmore-based models, to reduce calculation time. However, such models are not accurate enough when characterizing the high speed on-chip interconnects.

Considering the difficulties existing in current methods and needs for new methods, this thesis proposes a new analytic method based on the numerical inversion of the Laplace transform (NILT) to estimate the behavior of the on-chip interconnect in time domain. Compared to the existing methods, the proposed method can provide accurate and efficient estimations for electrically long interconnect line and for input signals with very sharper rise time, which describes how long the input signal spends in the intermediate state between two valid logic levels. Numerical examples verify the accuracy and efficiency of the proposed method when comparing to two popularly existing methods.

Acknowledgments

I would like to acknowledge sincerely towards Professor Peter J. Simpson, Professor Abdallah Shami, Professor Kazimierz Adamiak, Professor Mehrdad R. Kermani, Professor Serguei Primak, Stephanie Tigert for their important help and support. In addition, I would like to thank every student and friend of the Department of Electrical and Computer Engineering, University of Western Ontario for their friendly and supportive manner. I would like to extend my thanks towards my colleagues at Western University, for their continuous help and useful discussions.

Lastly, I wish to express a special thanks to my son, for his endless support, strong encouragement and selfless love.

Contents

List of Figures

List of Tables

Abbreviations

VLSI	Very-Large-Scale-Integration
IC	Integrated Circuits
SI	Signal Integrity
MTLs	Multiconductor Transmission Lines
RC	Resistive-Capacitive
RLC	Resistive-Inductive-Capacitive
CAD	Computer Aided Design
SPICE	Simulation Program with Integrated Circuit Emphasis
CPU	Central Processing Unit
PRIMA	Passive Reduced-order Interconnect Macromodeling Algorithm
MNA	Modified Nodal Analysis
ODE	Ordinary Differential Equations
PDE	Partial Differential Equations
NILT	Numerical Inversion of the Laplace Transform
VDSM	Very Deep Submicrometer
TEM	Transverse Electromagnetic
MOR	Model Order Reduction
AWE	Asymptotic Waveform Evaluation

WR	Waveform Relaxation
LP	Longitudinal Partitioning
LMS	Linear Multi-Step

Chapter 1

Introduction

1.1 Background and Motivation

Very-large-scale-integration (VLSI) is referred as the process, by which the integrated circuits (IC) are created combining billions of transistors into a single chip [1], [2]. Presently, VLSI technology and IC chips have been widely used in such fields as mobile, satellite communication, computer hardware, microelectromechanical systems, robotics and other electronic systems. The rapid decrease in feature size, associated growth in circuit complexity and density, and coupled with higher operating speeds, has made the analysis of on-chip interconnects a critical aspect of system reliability, speed of operation, and cost [3]. At gigahertz range, the design of clocks, frequencies, have become very critical when determining the operation speed of circuits [4]. Furthermore, as the frequency increases, interconnects behave like transmission lines, which play an important role in the majority of signal degradation and speed performance in high-speed systems [5].

In an IC, the electrical signal speed is mainly determined by two factors. The first factor is the switching time of the individual transistor, which is known as transistor gate delay. The second factor is the signal propagation delay occurred between transistors, which is known as the interconnect delay or the wire delay. Currently, the overall circuit performance depends mostly on the delay of interconnects rather than the delay of devices [6], [7]. The interconnect delay may result in improper triggering and timing uncertainty if it is not effectively quantified. Due to the improperly designed interconnects, the interconnect effects such as the signal delay, reflection, dispersion, distortion, crosstalk noise, signal overshoot, attenuation and ringing can severely degrade signal integrity and cause false actions of the circuits. As a result, circuit designers must consider the effects of interconnects at the early stages of the design cycle to guarantee system performance and reliability and to avoid costly redesigns [8], [9].

Evaluating the time domain response for lossy multiconductor transmission lines (MTLs) is greatly important for characterizing high speed interconnections in the design

of digital computers and communication systems [10]. In the past, on-chip interconnects at relatively low frequencies were modeled as a single lumped capacitance, and then the lumped resistance-capacitance and even lumped resistance-capacitance-inductance models were presented when analyzing the performance of on-chip interconnects [11], [12]. In current integrated circuit designs, wire inductance at higher operating speeds can no longer be ignored due to the longer electrical line lengths, which become a significant fraction of the fundamental and harmonic wavelength of the transient signal [13]. The dominant inductive effects are the key contributor to signal degradation, such as ringing and nonmonotonic response contaminated with spurious glitches on active lines. Also, the dominant inductive effects cause increase of line propagation delay. The conventional lumped interconnect models are not adequate when characterizing the performance of interconnect. Thus, an analytic *RLC* interconnect model, which is highly accurate, speedy and stable for analyzing high speed signals propagation in interconnects, are required to efficiently characterize the signal responses of today's high-performance integrated circuits and to guarantee the signal integrity.

On the other hand, the significant growth in the scale of integrated circuits being designed in VLSI has generated the need for new method of circuit simulation. Analyses of on-chip interconnects are based on either simulation techniques or closed-form analytic formulas. Computer aided design (CAD) tools have played an important role in signal integrity simulation by reducing time consuming and providing efficient simulation technique [14]. However, sophisticated CAD tools are required to solve myriad of problems for interconnect simulation. One most common used method for time domain response analysis of the large distributed interconnects is commercial circuit simulators with integrated circuit emphasis such as SPICE, which uses numerical integration or convolution techniques to provide accurate results [15]. Various SPICE models have been used to represent the transmission line segments such as the conventional lumped model [16], [17], W-element [18] and delay extraction based passive compact transmission line (DEPAC) model [19]. However, irrespective of the nature of the model involved, such models typically result in large CPU costs and storage to be used in early layout optimization design [20].

For an iterative layout design of densely populated integrated circuits composed of billions of transistors, efficient and accurate analytic models are needed to predict the delay and rise times of interconnect circuits [20]-[40]. In the past, on-chip interconnects were modeled as resistive-capacitive (*RC*) lines and single-pole Elmore-based models [21], [22] were most widely used to estimate signal delay. In current integrated circuit designs, wire inductance can no longer be ignored due to higher operating speeds causing longer electrical line lengths. Thus, analytic *RLC* interconnect models are required to efficiently characterize the signal responses of today's high-performance integrated circuits [23].

The issue of developing fast analytic *RLC* interconnect models has been an active area of research [20]-[40]. To capture the nonmonotonic response of *RLC* lines, Elmore-based models have been extended to two-pole transfer functions [1], [16], [24], [25]. However, the accuracy of these models is limited since two poles may not be enough to capture the signal delay and high frequency effects of inductive dominant *RLC* lines. Furthermore, the two-pole models of [1], [16], [24], [25], explicitly calculate the moments and extending these methods to higher orders is challenging since explicit moment matching techniques can produce unstable poles for higher order rational approximations [26]-[28]. To obtain stable higher order rational approximations, techniques such as passive reduced-order interconnect macromodeling algorithm (PRIMA) [29], Padé via Lanczos (PVL) [30] and Arnoldi algorithm [31], [32] are proposed. These methods use implicit moment matching techniques on lumped *RLC* circuits to obtain a low order rational approximation. Generally, these techniques can obtain higher order rational approximations at the expense of computational complexity since using lumped *RLC* circuits to approximate interconnects increases the size of the modified nodal analysis (MNA) matrices which increases the complexity of calculating the moments. In [33], an *RLC* interconnect model based on a Fourier series is proposed which directly uses the transfer function of interconnects without any approximation. This method is suitable for finding the steady state solution of periodic signals but is not directly applicable for transient analysis. Alternatively, to capture the long signal delay of inductive dominant *RLC* interconnects, methodologies based on traveling wave [34], Bessel functions [35]-[38] and delay algebraic equations [39] are developed. However, implementing these techniques to *RLC* tree circuits is a challenging task.

In addition, analysis of distributed transmission lines terminated with nonlinear elements causes mixed frequency/time problem [41]. This problem arises from the fact that circuit simulators solve time domain analysis using ordinary differential equations (ODE), while the transmission lines are traditionally characterized and solved in the frequency-domain using partial differential equations (PDE). It is impractical when using a single method to analyze the entire system which contains both linear and nonlinear circuits. Consequently, it is necessary to develop an effective method, which can provide the accurate and efficient time domain analysis for large distributed interconnects with arbitrary terminations such that the entire system can be divided into enumerated subsystems and each subsystem can be solved separately in the efficient way. Then the accurate time domain solution of the entire system will be obtained by solving the decoupled subsystems. Hence, efficient algorithm for the time domain response analysis of high speed on-chip distributed interconnect networks with arbitrary topologies, sizes and terminations are warranted [42].

The two-pole model in [25] is presented to simulate the distributed interconnect tree structure. However, it may generate unstable poles when increasing the approximation order. As an advanced moment matching technique, passive reduced-order interconnect macromodeling algorithm (PRIMA) is first presented in [29] to meet the challenge occurred in [25]. Though higher order rational approximations can be obtained by PRIMA [29], the computational complexity is greatly increased. In this work, an analytic method for signal delay evaluation for on-chip *RLC* distributed interconnects is proposed, by which arbitrary accuracy can be obtained by increasing approximation orders but without causing numerical instable issue occurred in [25]. In addition, the proposed method does not need to divide each transmission line into sections, but directly uses the frequency solution of interconnect circuits for approximating the transfer function of interconnects. As a result, the proposed method has the obvious advantages over the techniques in [25] and [29] when considering the accuracy and efficiency. In this thesis, the proposed method is verified in terms of accuracy and efficiency when using it to calculate signal delays at certain nodes and then comparing the calculated results with those obtained using the two-pole model [25] and PRIMA [29], respectively. Consequently, comparisons are performed only

considering the essential principles of [25] and [29], which firstly proposed the two-pole model and PRIMA, respectively. Thus, the goal of this thesis is to address effectively the shortcomings of the existing techniques by proposing a new analytic method for on-chip *RLC* distributed interconnects to evaluate signal delay. In specific, the contributions of the thesis are listed below.

1.2 Contributions

The objective of this thesis is the development of an analytic delay model of RLC interconnect networks using numerical inversion of the Laplace transform (NILT). The method proposed here provides an accurate, efficient and numerically stable approach for signal delay estimations, which includes the estimations for 10%, 50% and 90% delays. The proposed method is suitable for transient analysis. Since the integration formula of NILT is numerically stable for higher order approximations, the developed algorithm provides a mechanism to increase the accuracy of the delay estimates for cases when inductive effects are significant, the length of the line increases, or when the rise time of the signal becomes sharper. The main contributions of this thesis ae as follows.

1. The exact transfer function at the far end is derived first based on the ABCD matrix for distributed *RLC* transmission lines. Then the derivation of transfer function is extended to arbitrary interconnect tree structure by considering some essential parameters such as the propagation constant, the characteristic impedance, and the equivalent input impedance and load. The transfer function at any node of the interconnect structure can be solved exactly, by which the exact frequency domain response at any required node is possible once the input signal is provided [25], [33].

2. An accurate and efficient analytic delay model is proposed for on-chip *RLC* interconnects. The proposed algorithm is based on the numerical inversion of the Laplace transform (NILT) which was previously used as an integration formula for SPICE analysis of interconnect circuits [43]-[47]. The NILT based proposed method directly uses the frequency solution of interconnect circuits without using macromodeling algorithms (i.e. such as lumped *RLC* circuits [47]) or moment

matching techniques [26]-[32] to approximate the transfer function of interconnects. The proposed method does not require determination of the poles and residues of the function under consideration, conversely, it only needs the computation of the frequency domain function at preassigned complex points and forming a weighted sum.

3. The proposed method can exactly invert a certain number of terms of the Taylor expansion of the time domain response, and it is thus equivalent to an integration formula that is stable for higher order approximations [43], [45]. In addition, the proposed algorithm provides a mechanism to increase the accuracy of the model for electrically long *RLC* interconnect circuits. Furthermore, the higher order time functions obtained by NILT can be solved in parallel using multi-core processing techniques.

4. The proposed model is theoretically stable and computationally efficient. In order to verify the excellent accuracy and efficiency of the proposed method, time domain responses and signal delays including 10%, 50% and 90% delays are calculated for on-chip *RLC* distributed interconnect structure ranging from a single transmission line to complicated tree network. As well the results are compared to those of HSPICE, the two-pole model [25] and PRIMA [29] with unit step and rising unit ramp inputs, respectively.

1.3 Organization of the Thesis

The organization of the thesis is as follows. Chapter 2 briefly discusses the general categories of interconnect models, and presents the derivation of transfer function for single transmission line and interconnect tree structure, respectively. The chapter also reviews some existing relevant techniques for modeling interconnects. In Chapter 3, the mathematical basis for the proposed method is prescribed first. Then a detailed description for developing the proposed method is provided. In addition, the significant properties and application of the proposed method are presented. Numerical examples are given to demonstrate the accuracy and efficiency of the proposed method in Chapter 4. Finally, Chapter 5 presents the conclusions and proposed future research.

Chapter 2

Overview of Interconnects Modeling and Simulation Techniques

2.1 Introduction

Interconnects are a dominant factor in determining the performance, reliability and cost for the overall circuit due to the advancements in high-speed very deep submicrometer (VDSM) VLSI designs and particularly nanosystems. As frequency increases, interconnects gradually display resistive, capacitive and inductive effects, which can severely degrade the signal integrity of networks. As VLSI technology progresses, the signal integrity is becoming more important since the dominant signal distortion and logical failures are not caused by logic gates but by the interconnect lines. For interconnects, as the electrical conducting structures, which propagate signals, the accurate modeling, analysis and characterization of the interconnects are essential for modern VLSI circuit design nowadays. Selecting a suitable model and solution algorithm depends on the operating frequency, physical structure of interconnects and practical application. The goal of this chapter is to review some of the methods used in the literature for interconnect analysis.

This chapter is organized as follows. Section 2.2 presents various transmission line models used for interconnect analysis. The frequency domain analysis and solution for single transmission line is described in Section 2.3. Some popular modeling and simulation techniques are provided in Section 2.4. Conclusion for this chapter is provided in Section 2.5.

2.2 Interconnect Models

The dramatical growth in density, operating speeds and complexity of modern integrated circuits has made interconnect analysis a requirement for all state-of-the-art circuit simulators. Interconnect exists at various levels of design hierarchy such as on-chip, packaging structures, multi-chip modules, printed-circuit-boards and backplanes shown as Figure 2.1. For many high-speed electrical networks, overall performance may depend

mostly on the delay of interconnects rather than the delay of devices. Consequently, designers must consider interconnect analysis at the early stages of the design cycle to ensure circuit performance and reliability.

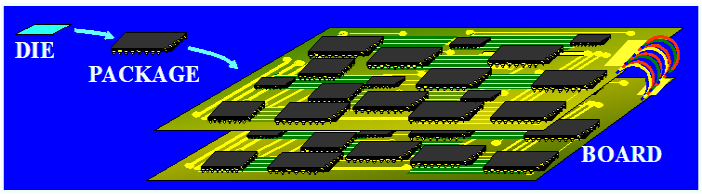


Figure 2. 1: Interconnect hierarchy

Electrical models are required in order to simulate interconnects with circuit elements. The interconnect model selection is determined by both the physical interconnect structure and the circuit operating frequency.

2.2.1 Quasi-Transverse Electromagnetic Models

Interconnects with homogeneous mediums and perfect conductors produce TEM (Transverse electromagnetic) waves, which is a mode of propagation where the electric and magnetic field lines are transverse (i.e. perpendicular to one another and to the direction of propagation) [41]. Electromagnetic waves with many velocities are produced when interconnect is with inhomogeneous mediums and an electric field along the conductor surface is generated when interconnect has imperfect conductors, which both violate the TEM characteristics due to the fact that there is only one velocity when TEM waves propagate and there is no electric field along the conductor surface. Nevertheless, this generated structure is similar to the TEM structure and can be approximated by TEM waves as quasi-TEM waves by ignoring the component of the mutually transverse electric and magnetic fields along the line axis [41]. The voltages and currents for the quasi-TEM distributed models are described by partial differential equations (PDEs) known as Telegrapher's equations.

At higher frequencies, edge, proximity, and skin effects become prominent and distributed models with frequency-dependent parameters are required. The difficulty with quasi-TEM distributed models is that they cannot be directly linked to circuit simulators

such as SPICE, which solve nonlinear ordinary differential equations (ODEs) and have difficulty solving the PDEs. To overcome this difficulty, numerical techniques are used to convert distributed models into ODEs.

2.2.2 Full Wave Models

The field components in the direction of propagation can no longer be ignored and quasi-TEM assumption become inadequate in describing interconnects when the cross-sectional dimensions of interconnects become a significant fraction of the circuit's operating wavelength. Under these conditions, full wave models are required, which are able to account for all possible field components and satisfy all boundary conditions required to accurately model the high-frequency effects of interconnects [48], [49].

2.2.3 Quasi-TEM Models vs. Full Wave Models

Quasi-TEM assumptions remain the dominant trend for analysing lossy MTLs, since the approximation is valid for most practical interconnect structures and offers relative ease. Full wave models usually produce large set of equations and are very CPU intensive to solve when compared to quasi-TEM models.

2.3 Frequency Domain Analysis

2.3.1 Telegrapher's Equations

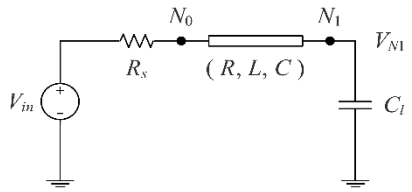


Figure 2. 2: Circuit model for a single distributed *RLC* interconnect

A classical interconnect model is shown in Figure 2.2 represented by a distributed *RLC* transmission line, where *l* is the length of transmission line, *R*, *L*, and *C* are the per-unit-length (p.u.l) resistance (Ω/m), inductance (H/m), and capacitance (C/m) of the transmission line, respectively. The model in Figure 2.2 represents a point-to-point interconnection driven by a transistor linearized as voltage source V_{in} serially connected

with a linear driver resistance R_s and connected to the next gate, which is modeled as a load capacitance C_l [20]-[25], [33]-[38]. Analyzing on-chip *RLC* interconnects in frequency domain starts with Telegrapher's equations, which are a pair of linear differential equations used to describe the voltage and current along the transmission line. For transmission line, the voltage and current are functions of position *x* and time *t*. These equations are

$$\frac{\partial}{\partial x}V(x,s) = -(R + sL)I(x,s)$$

$$\frac{\partial}{\partial x}I(x,s) = -sCV(x,s) \tag{2.1}$$

where *s* is the Laplace operator, *x* is the position variable; $V(x, s)$ and $I(x, s)$ are the voltage and current at the position *x* along the transmission line in frequency domain; Let

$$Z = R + sL$$

$$Y = sC \tag{2.2}$$

(2.1) can be rewritten as

$$\frac{\partial}{\partial x}V(x,s) = -ZI(x,s)$$

$$\frac{\partial}{\partial x}I(x,s) = -YV(x,s) \tag{2.3}$$

Using the exponential matrix, the solution of (2.3) can be expressed as

$$\begin{bmatrix} V(l,s) \\ -I(l,s) \end{bmatrix} = e^{\Phi l}\begin{bmatrix} V(0,s) \\ I(0,s) \end{bmatrix} \tag{2.4}$$

where

$$\Phi = \begin{bmatrix} 0 & -Z \\ -Y & 0 \end{bmatrix} \tag{2.5}$$

2.3.2 Transfer Function in Frequency Domain

The expression of (2.4) is in terms of ABCD-parameters, which can be expressed using the cosh and sinh functions as shown below,

$$\begin{bmatrix} V(l,s) \\ -I(l,s) \end{bmatrix} = \begin{bmatrix} \cosh(\gamma) & -Z_c\sinh(\gamma) \\ -\frac{1}{Z_c}\sinh(\gamma) & \cosh(\gamma) \end{bmatrix} \begin{bmatrix} V(0,s) \\ I(0,s) \end{bmatrix} \tag{2.6}$$

where $\gamma = l\sqrt{ZY}$, $Z_c = \sqrt{\frac{Z}{Y}}$. The expansions of ABCD-parameters for a distributed *RLC* transmission line are

$$\cosh(\gamma) = 1 + \frac{1}{2!}l^2RCs + \left(\frac{1}{2!}l^2LC + \frac{1}{4!}l^4R^2C^2\right)s^2 + \cdots$$

$$Z_c\sinh(\gamma) = Rl + \left(Ll + \frac{1}{3!}R^2Cl^3\right)s + \left(\frac{2}{3!}RLCl^3 + \frac{1}{5!}R^3C^2l^5\right)s^2 + \cdots$$

$$\frac{1}{Z_c}\sinh(\gamma) = Cls + \frac{1}{3!}l^3RC^2s^2 + \cdots \tag{2.7}$$

Considering the equivalent circuit shown as in Figure 2.2, the boundary conditions can be expressed as

$$V_{in} = V(0,s) + R_SI(0,s) \tag{2.8}$$

$$V(l,s) = -sC_l\, I(l,s) \tag{2.9}$$

From (2.6) to (2.9), the transfer function from the input to the far end of the transmission can be described as [33]

$$H(s) = \frac{1}{(1 + sR_sC_l)\cosh(\gamma) + (R_s/Z_c + sC_lZ_c)\sinh(\gamma)} \tag{2.10}$$

The frequency-domain solution at the far end is expressed as

$$V_{N_1}(s) = \frac{V_{in}}{(1 + sR_sC_l)\cosh(\gamma) + (R_s/Z_c + sC_lZ_c)\sinh(\gamma)} \tag{2.11}$$

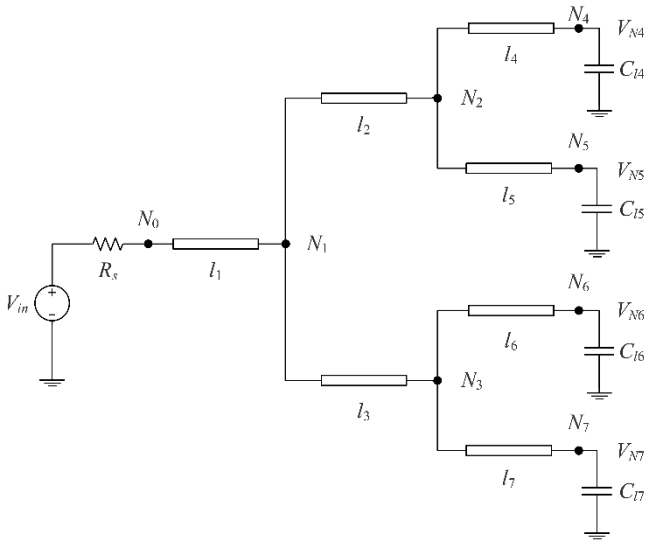


Figure 2. 3: General distributed *RCL*

Interconnect tree shown as in Figure 2.3 is widely used in clock distribution networks. In such a structure, a driver with an output resistance R_s is connected to the root of the tree N_0. All of the output nodes ($N_4 \ldots N_7$) are called leaves and connected with load buffers which can be used to drive the *RLC* trees in the next level. The load buffers are modeled by capacitors. All of the branches in the tree are represented by distributed *RLC* lines. The tree can be balanced or unbalanced; however, unbalanced tree exhibits more complex characteristics than balanced trees [25], [33].

The transfer function from node N_0 to a certain node N_i is defined as the product of the transfer function of all of the branches along the unique path from N_0 to N_i. For a transmission line of length l with load Z_L at the far end, the input impedance seen from the near end is [33]

$$Z_{in} = Z_c \frac{Z_L + Z_c \tanh(\gamma)}{Z_c + Z_L \tanh(\gamma)} \tag{2.12}$$

If a node branches out to multiple interconnects (such as node N_1 of Figure 2.3), then the load impedance seen at this node is the parallel combination of the input impedance of the downstream branches connected to this node [33]. Hence, the transfer function from

the voltage source to a certain node N_i is expressed as [33]

$$H_i(s) = \frac{Z_{L,0}}{R_s + Z_{L,0}} \prod_k \frac{1}{\cosh(\gamma_k) + \left(Z_{c,k}/Z_{L,k}\right)\sinh(\gamma_k)} \tag{2.13}$$

where k is the index following each branch in the path from node N_0 to N_i. γ_k and $Z_{c,k}$ are the propagation operator and characteristic impedance of the k^{th} line, respectively; $Z_{L,0}$ is the input impedance seen from node N_0. $Z_{L,k}$ is the load impedance observed at node N_k.

The voltage V_{N_i} at node N_i is

$$V_{N_i}(s) = \frac{V_{in} Z_{L,0}}{R_s + Z_{L,0}} \prod_k \frac{1}{\cosh(\gamma_k) + \left(Z_{c,k}/Z_{L,k}\right)\sinh(\gamma_k)} \tag{2.14}$$

In this work, the input voltage for analyzing (2.11) and (2.14) is a step response $V_{in}(s) = V_{DD}/s$ or a ramp response $V_{in}(s) = V_{DD}(1 - e^{-st_r})/s^2 t_r$, where V_{DD} is the voltage amplitude, and t_r is the rise time of the ramp input. Equations (2.11) and (2.14) do not have a direct representation in the time-domain which makes it difficult to analytically predict the 50% signal delay and rise time of voltage signals. In [12], [22], [25], [33], [50]-[52] various closed form models have been developed to provide efficient expressions for (2.11) and (2.14) in time domain.

2.4 Modeling and Simulation Techniques Review

2.4.1 General Class of Analysis Techniques

Either simulation techniques or closed form analytic formulas are required in analyzing on-chip interconnects. As discussed in Section 2.3, the distributed interconnects are best described by Telegrapher's equations in the form of partial differential equations, which provides an exact transfer function in the frequency domain. However, it has difficulty in providing an exact time domain expression due to the complicated hyperbolic functions of complex frequency *s*. Consequently, the most important difficulty is the numerical integration problem when modeling on-chip interconnects for signal integrity verification. Simulation tools such as SPICE employs numerical integration or convolution techniques

to provide accurate results at each time step. However, these simulation techniques are computationally expensive in the application of layout optimization [52].

To accurately estimate the signal delays of interconnects, accurate and efficient analytic models are required for an iterative layout design of densely populated integrated circuits composed of billions of gates. Since poles and residues have direct representations in time domain, the frequency domain transfer function of interconnect can be expressed as a simple rational function represented in the form of poles and residues according to the conventional methods [20]-[22], [34], [51]. In this manner, the time domain response of interconnect can be estimated at each time step without using numerical integration technique. Inspired by this idea, the Elmore delay based single pole *RC* model is presented in [11], [21], [22], [50] to estimate signal delay at early stage for on-chip interconnect. As the advancement of integrated circuit design, line inductance effect becomes significant due to the higher operating speeds and electrically long line. Hence, more accurate and efficient model is needed for covering the characteristics of high performance of today's integrated circuits. Compared to the simulation tools, such as SPICE, closed-form analytic models are simpler while preserving reasonable accuracy.

2.4.2 Moment Matching Techniques Based on Model Order Reduction Algorithm

The poles, which have significant effects on the characteristics of system, are defined the dominant poles. For large linear systems, they usually have large number of poles and only small percentage of these poles are dominant, which forms the underlying concept of MOR (Model Order Reduction). MOR algorithm is capable of capturing the frequency response of large linear systems using low-order rational approximations [29]. By capturing only the dominant poles, the computational complexity and large size of the simulation can be significantly reduced while keeping accuracy uncompromised.

MOR algorithm can be in general classified two main categories: explicit moment matching technique based on direct Padé approximations and implicit moment matching technique based on projecting large matrices on its dominant eigenspace [26].

2.4.2.1 Explicit Moment Matching

Explicit moment matching technique [26] uses direct Padé approximations to capture the frequency impulse response of network. In order to illustrate the concept of explicit moment matching technique, three different interconnect models are considered in this section.

A. Single Pole Lumped Model

[21] presented the earliest Elmore delay based model used for the transient response of damped linear network, which is one of the most popular models for SI verification in interconnects and widely used in VLSI design theory [53]-[55]. In this model, each interconnect is modeled as simple lumped RC circuits without loss of generality shown as in Figure 2.4.

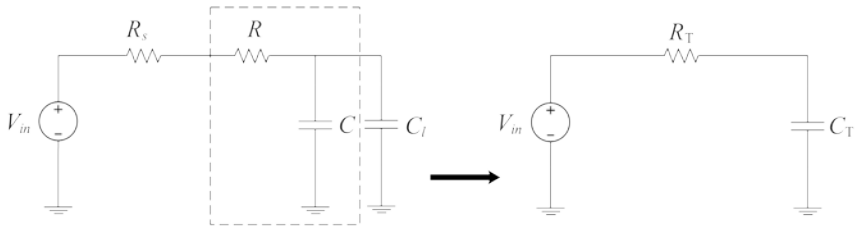


Figure 2. 4: Circuit model of Elmore *RC* interconnect

The transfer function for the circuit in Figure 2.4 can be expressed as

$$H(s) = \frac{1}{(1 + sR_TC_T)} \tag{2.15}$$

where $R_T = R_s + R$ is the total interconnect resistance including source resistance and $C_T = C_l + C$ is the total interconnect capacitance including load capacitance, respectively. Suppose a unit step input is provided, the output response in time domain is given as

$$V_{out}(t) = \left(1 - exp(-t/T_D)\right) \tag{2.16}$$

where $T_D = R_TC_T$ is defined as the time constant.

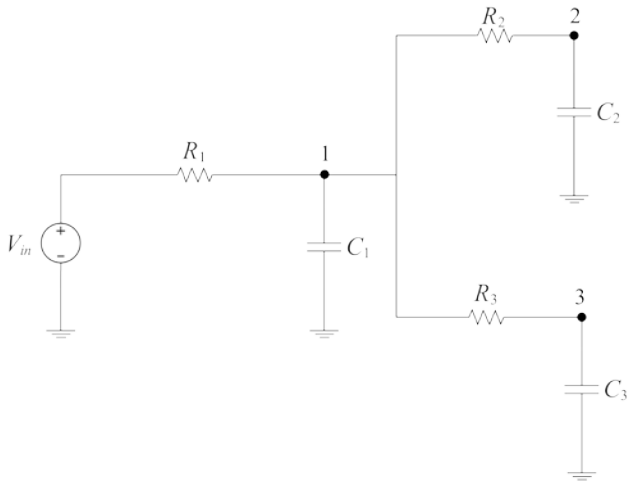


Figure 2. 5: Circuit model of Elmore *RC* tree interconnect

In a VLSI circuit, an interconnect line is in general a tree rather than a single line. Hence, characterizing signal waveforms in a tree interconnect structure is primarily important. Within industry Elmore delay model is one of the most popular delay models used for *RC* tree interconnects, in which each interconnect is considered as a lumped *RC* circuit depicted in Figure 2.5. The introduction of a simple closed form solution for the time constant T_D makes the Elmore delay model appealing for *RC* tree interconnects. For the *RC* tree interconnect shown in Figure 2.5, the time constant T_{Di} at node *i* can be represented as

$$T_{Di} = \sum_{k} C_k R_{ik} \tag{2.17}$$

where *k* is an index that covers each capacitor in the circuit, R_k is the common resistance from the input to the node *i* and *k*. The approximated transfer function at node *i* for the structure in Figure 2.5 is represented using a first-order (single pole) [23], [24]

$$H_i(s) = \left(1 + s \sum_{k} C_k R_{ik}\right)^{-1} \tag{2.18}$$

The first-order approximation (2.18) matches the first moment of the transfer function at node *i* but approximates the higher-order moments by

$$m_r^i = \left(-\sum_k C_k R_{ik} \right)^r \tag{2.19}$$

Seen in (2.18)

$$H_i(s) = 1 + m_1^i s + m_2^i s^2 + \cdots = 1 - s\sum_k C_k R_{ik} + s^2 \left(\sum_k C_k R_{ik} \right)^2 - \cdots \tag{2.20}$$

Using one moment matching, the step response of *RC* tree interconnects is $V(t) = 1 - e^{t/T_D}$. Elmore delay is the 50% propagation delay, which is expressed as

$$t_{50\%} = 0.69 T_D = 0.69 \sum_k C_k R_{ik} \tag{2.21}$$

Elmore delay based single pole model can provide the simple and useful analytical formula for lumped *RC* tree interconnects. In addition, Elmore delay model can evaluate computationally faster and always leads to stable solution. Thus, it is still widely used within industry. However, Elmore delay model cannot provide an equivalent delay model for *RLC* tree interconnects primarily because the Elmore delay does not properly cover nonmonotone responses which are caused by large line inductance. The inductance effect occurs and becomes obvious in *RLC* circuits when modern switching speeds touched the GHz range. As a result, at least a second-order approximation is required to properly characterize the nonmonotone response.

B. Two Pole Lumped Model

As discussed previously, Elmore delay based single pole model fail to properly represent the *RCL* interconnects due to the nonmonotone response. In addition, equations of (2.11) and (2.14) contain complicated hyperbolic functions of complex frequency variable *s*, which makes (2.11) and (2.14) do not have direct expressions in time domain. Thus, it is difficult to analytically estimate the signal delays for *RLC* interconnects. To solve this problem, [24] develops the equivalent two-pole Elmore delay model for *RLC* tree interconnects, in which a single transmission line is modeled as a lumped *RLC* network

shown as Figure 2.6,

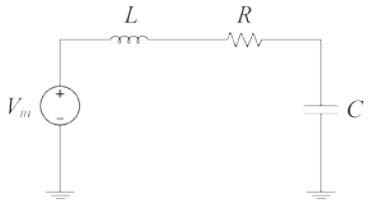


Figure 2. 6: Simple *RLC* circuit

The circuit in Figure 2.6 can be represented using a second order transfer function [24]

$$H(s) = \frac{1}{s^2LC + sRC + 1} \tag{2.22}$$

In (2.22), the coefficient of s^1 is *RC*, which is the coefficient of the Elmore time constant, is independent of inductance *L*. However, the inductance *L* can affect the circuit response significantly. Reconfiguring the transfer function (2.22) for the circuit in Figure 2.6 and explaining the inductance effects on circuit responses as follows [24],

$$H(s) = \frac{\omega_n^2}{s^2 + 2s\zeta\omega_n + \omega_n^2} \tag{2.23}$$

where

$$\zeta = \frac{RC}{2\sqrt{LC}} \tag{2.24}$$

$$\omega_n = \frac{1}{\sqrt{LC}} \tag{2.25}$$

The poles of transfer function (2.23) are

$$P_{1,2} = \omega_n\left(-\zeta \pm \sqrt{\zeta^2 - 1}\right) \tag{2.26}$$

As observed in (2.24), ζ decreases as the inductance *L* increases. Thus, it is necessary to use a second order transfer function as (2.23) to properly describe the nonmonotone

response of an *RLC* circuit. Expending the transfer function of (2.23) as

$$H(s) = 1 + m_1 s + m_2 s^2 + \cdots = 1 - s\left(\frac{2\zeta}{\omega_n}\right) + s^2\left(\frac{-1 + (2\zeta)^2}{\omega_n^2}\right) - \cdots \quad (2.27)$$

In (2.27), parameters ζ and ω_n can be calculated using moments,

$$\zeta = -\frac{m_1}{2}\frac{1}{\sqrt{m_1^2 - m_2}} \quad (2.28)$$

$$\omega_n = \frac{1}{\sqrt{m_1^2 - m_2}} \quad (2.29)$$

Let

$$b_1 = -m_1 \quad (2.30)$$

$$b_2 = m_1^2 - m_2 \quad (2.31)$$

and

$$\Delta = b_1^2 - 4b_2 \quad (2.32)$$

Rewriting the poles of (2.26) in terms of variables $b_1$and b_2,

$$P_{1,2} = \left(-b_1 \pm \sqrt{b_1^2 - 4b_2}\right)/2b_2 \quad (2.33)$$

For a step input with supply voltage of V_D, the time domain response of the network can be determined using the second order approximation of transfer function as (2.23). If ζ is greater than one, which leads to $\Delta > 0$, the poles of (2.33) are real and monotone response occurs as

$$V_{out}(t) = V_D\left(1 - \frac{P_2}{P_2 - P_1}e^{P_1 t} + \frac{P_1}{P_2 - P_1}e^{P_2 t}\right) \quad (2.34)$$

If ζ is equal to one, which leads to $\Delta = 0$, two poles of (2.33) are equal and the response is

a critically damped response,

$$V_{out}(t) = V_D\left(1 - \left(1 + \frac{2t}{b_1}\right)e^{-(b_1/2b_2)t}\right) \tag{2.35}$$

If ζ is less than one, which leads to $\Delta < 0$, the poles of (2.33) are complex and nonmonotone response occurs,

$$V_{out}(t) = V_D\left(1 - \frac{\sqrt{\alpha^2 + \beta^2}}{\beta}e^{-\alpha t}.\sin(\beta t + \rho)\right) \tag{2.36}$$

where $\alpha = b_1/2b_2$, $\beta = \sqrt{4b_2 - b_1^2}/2b_2$ and $\rho = \tan^{-1}(\beta/\alpha)$ [25]. Hence, the propagation delays can be calculated using (2.34) - (2.36).

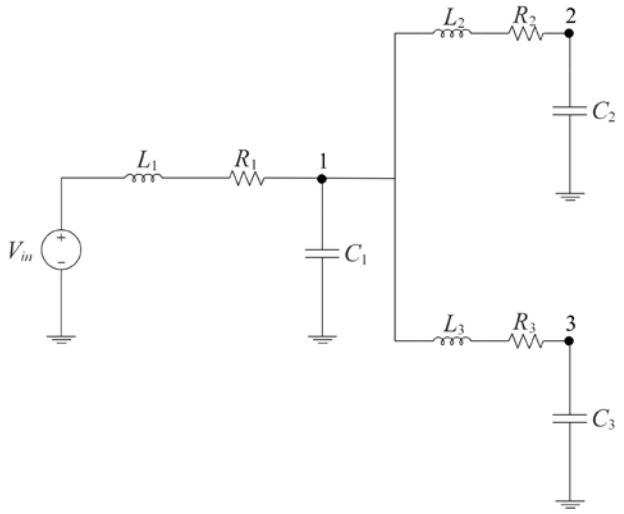


Figure 2. 7: General model of lumped *RLC* tree

Extending the single lumped *RLC* line interconnect to the lumped *RLC* tree interconnect depicted as Figure 2.7, the voltage drop at node *i* can be expressed as

$$V_i(s) = V_{in}(s) - \sum_k sC_kV_k(s)(R_{ki} + sL_{ki}) \tag{2.37}$$

For the unit impulse input, $V_{in}(s) = 1$ and the nodal voltages of the tree structure are the unit impulse response of these nodes. As a result, $V_i(s)$ provides the normalized transfer

function $H_i(s)$ at node *i* as

$$H_i(s) = 1 + m_1^i s + m_2^i s^2 + \cdots = 1 - \sum_k sC_k V_k(s)(R_{ki} + sL_{ki}) \tag{2.38}$$

The first and second moments at node *i* can be represented as [24]

$$m_1^i = -\sum_k C_k R_{ik} \tag{2.39}$$

and

$$m_2^i = \left(\sum_k C_k R_{ik}\right)^2 - \sum_k C_k L_{ik} \tag{2.40}$$

Considering (2.28) and (2.29), the variables ζ_i and ω_{ni} that are used to express the second order approximation of the transfer function at node *i* are

$$\zeta_i = \frac{1}{2}\left(\sum_k C_k R_{ik} \Big/ \sqrt{\sum_k C_k L_{ik}}\right) \tag{2.41}$$

$$\omega_{ni} = 1 \Big/ \sqrt{\sum_k C_k L_{ik}} \tag{2.42}$$

Once the input signal in time domain is provided, the frequency domain response at node *i* of the tree structure can be calculated by multiplying the Laplace transform of the input signal with the second order approximation of the transfer function described by (2.23), (2.41) and (2.42). The resulting expression of the time domain response can be obtained after applying the inverse Laplace transform. For a step input with a supply voltage V_{DD}, the time domain response at node *i* can be written as,

$$V_i(t) = V_{DD} + \frac{V_{DD}}{2\sqrt{\zeta_i^2 - 1}}\left(\frac{e^{\omega_{ni}t\left(-\zeta_i + \sqrt{\zeta_i^2-1}\right)}}{-\zeta_i + \sqrt{\zeta_i^2-1}} - \frac{e^{\omega_{ni}t\left(-\zeta_i - \sqrt{\zeta_i^2-1}\right)}}{-\zeta_i - \sqrt{\zeta_i^2-1}}\right) \tag{2.43}$$

In (2.43), the time t is scaled by ω_{ni}. Let $\bar{t} = \omega_{ni}t$, the step response at node i is a function of only variable ζ_i,

$$\overline{V_i(t)} = V_{DD} + \frac{V_{DD}}{2\sqrt{\zeta_i^2-1}}\left(\frac{e^{\bar{t}\left(-\zeta_i+\sqrt{\zeta_i^2-1}\right)}}{-\zeta_i+\sqrt{\zeta_i^2-1}} - \frac{e^{\bar{t}\left(-\zeta_i-\sqrt{\zeta_i^2-1}\right)}}{-\zeta_i-\sqrt{\zeta_i^2-1}}\right) \tag{2.44}$$

where $\overline{V_i(t)}$ is the time scaled response at node i. Then the time scaled 50% delay $\overline{t_{50\%}}$and rise time $\overline{t_{rise}}$ at node i, which are the functions of only ζ_i, can be obtained using a curve fitting method [24],

$$\overline{t_{50\%}} = 1.047e^{(\zeta_i/0.85)} + 1.39\zeta_i \tag{2.45}$$

$$\overline{t_{rise}} = 6.017e^{(\zeta_i{}^{1.35}/0.4)} - 5e^{(\zeta_i{}^{1.25}/0.64)} + 4.39\zeta_i \tag{2.46}$$

Then the 50% delay and rise time at node i can be determined by $\overline{t_{50\%}}/\omega_{ni}$ and $\overline{t_{rise}}/\omega_{ni}$, respectively,

$$t_{50\%} = \left(1.047e^{(\zeta_i/0.85)} + 1.39\zeta_i\right)/\omega_{ni} \tag{2.47}$$

$$t_{rise} = \left(6.017e^{(\zeta_i{}^{1.35}/0.4)} - 5e^{(\zeta_i{}^{1.25}/0.64)} + 4.39\zeta_i\right)/\omega_{ni} \tag{2.48}$$

C. Two-Pole Distributed Model

For the circuit model in Figure 2.2, it is a linear time-invariant (LTI) system. Theoretically, the time domain response of the LTI system can be estimated by using the inverse of Laplace transform. As derived in (2.10) and (2.13) in Section 2.4.1, the transfer functions for a single *RLC* interconnect and *RLC* tree interconnect structure are complicated hyperbolic functions of complex frequency *s*. It is difficult to derive the inverse of Laplace transform directly for the transfer functions.

Based on the ABCD-parameters of a transmission line, the work of [25] improves the accuracy of the two-pole model by approximating the distributed hyperbolic functions of (2.10) and (2.13) as a power series (2.7) for fast simulation of distributed interconnect tree structure. Redrawing Figure 2.3 as Figure 2.8,

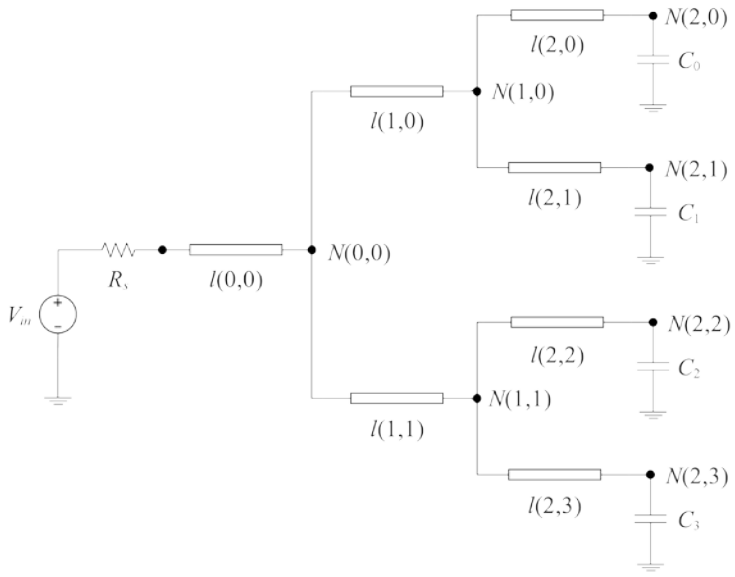


Figure 2. 8: General distributed *RLC* tree

The distributed *RLC* tree structure in Figure 2.8 has a level of n=2. Define i as the index of a level and the jth node in the ith level as $N(i, j)$, where i= 0, 1, …, n-1 and j= 0, 1, …, 2^i-1. In Figure 2.8, N (0,0) and $N(n, j)$ are root node and leaf node, respectively. C_j is denoted as the load capacitance of the leaf node $N(n, j)$. $T(i, j)$ is the interconnect between $N(i, j)$ and its ancestor [25]. For the given structure in Figure 2.8, each node $N(i, j)$ has both left sub-branch $T(i+1, 2j)$ between $N(i, j)$ and $N(i+1, 2j)$ and right sub-branch $T(i+1, 2j+1)$ between $N(i, j)$ and $N(i+1, 2j+1)$. Define $V_l(i, j)$ as the voltage of $N(i, j)$ at its left sub-branch $T(i+1, 2j)$, $V_r(i, j)$ as the voltage of $N(i, j)$ at its right sub-branch $T(i+1, 2j+1)$, and $V(i, j)$ as the voltage of $N(i, j)$ itself. Similarly, define $I_l(i, j)$ as the current of $T(i+1, 2j)$ at $N(i, j)$, $I_r(i, j)$ as the current of $T(i+1, 2j+1)$ at $N(i, j)$ and $I(i, j)$ as the current of $T(i, j)$ at $N(i, j)$.

Using ABCD-parameters to characterize interconnect $T(i, j)$,

$$\begin{bmatrix} V_l(i,j) \\ I_l(i,j) \end{bmatrix} = \begin{bmatrix} A(i+1,2j) & B(i+1,2j) \\ C(i+1,2j) & D(i+1,2j) \end{bmatrix} \begin{bmatrix} V(i+1,2j) \\ I(i+1,2j) \end{bmatrix} \tag{2.49}$$

$$\begin{bmatrix} V_r(i,j) \\ I_r(i,j) \end{bmatrix} = \begin{bmatrix} A(i+1,2j+1) & B(i+1,2j+1) \\ C(i+1,2j+1) & D(i+1,2j+1) \end{bmatrix} \begin{bmatrix} V(i+1,2j+1) \\ I(i+1,2j+1) \end{bmatrix} \tag{2.50}$$

Considering the terminal conditions,

$$I(n,j) = sC_j V(n,j) \tag{2.51}$$

and

$$V_{in} = A(0,0)V(0,0) + B(0,0)I(0,0) \tag{2.52}$$

Define $x_l(i,j), x_r(i,j), x(i,j)\ and\ z(i,j)$ as

$$x_l(i,j) = \frac{V_l(i,j)}{V(n,2^{n-1}j)} \tag{2.53}$$

$$x_r(i,j) = \frac{V_r(i,j)}{V\left(n,(2^{n-1}j + 2^{n-i-1})\right)} \tag{2.54}$$

$$x(i,j) = \frac{V(i,j)}{V(n,2^{n-1}j)} \tag{2.55}$$

$$z(i,j) = \frac{V\left(n,\left(2^{n-1}j + 2^{n-i-1}\right)\right)}{V(n,2^{n-1}j)} \tag{2.56}$$

where $x_l(i,j), x_r(i,j), x(i,j)$ represent the relationships between a branch node voltage and the corresponding leaf node voltage, and $z(i,j)$ represents the relationship between two leaf node voltages. For the root node, it satisfies

$$x(0,0) = \frac{V(0,0)}{V(n,0)} \tag{2.57}$$

For the leave node,

$$x(n,j) = 1 \tag{2.58}$$

Similarly, define $y_l(i,j), y_r(i,j), and\ y(i,j)$ as

$$y_l(i,j) = \frac{I_l(i,j)}{V(n, 2^{n-1}j)} \tag{2.59}$$

$$y_r(i,j) = \frac{I_r(i,j)}{V\left(n, (2^{n-1}j + 2^{n-i-1})\right)} \tag{2.60}$$

$$y(i,j) = \frac{I(i,j)}{V(n, 2^{n-1}j)} \tag{2.61}$$

where $y_l(i,j), y_r(i,j), and\ y(i,j)$ represent the relationships between a branch node current and the corresponding leaf node current. Considering (2.56) and (2.59) - (2.61), it holds

$$y(i,j) = y_l(i,j) + y_r(i,j)z(i,j) \tag{2.62}$$

For the root node,

$$y(0,0) = \frac{I(0,0)}{V(n,0)} \tag{2.63}$$

For the leaf nodes,

$$y(n,j) = sC_j \tag{2.64}$$

Considering (2.49), (2.50), (2.53), (2.54), (2.59) and (2.60), there are

$$\begin{bmatrix} x_l(i,j) \\ y_l(i,j) \end{bmatrix} = \begin{bmatrix} A(i+1,2j) & B(i+1,2j) \\ C(i+1,2j) & D(i+1,2j) \end{bmatrix} \begin{bmatrix} x(i+1,2j) \\ y(i+1,2j) \end{bmatrix} \tag{2.65}$$

$$\begin{bmatrix} x_r(i,j) \\ y_r(i,j) \end{bmatrix} = \begin{bmatrix} A(i+1,2j+1) & B(i+1,2j+1) \\ C(i+1,2j+1) & D(i+1,2j+1) \end{bmatrix} \begin{bmatrix} x(i+1,2j+1) \\ y(i+1,2j+1) \end{bmatrix} \tag{2.66}$$

The transfer function for the first leaf node $N(n,0)$ can be expressed as

$$H(s)|_0 = \frac{V(n,0)}{V_{in}} = \frac{1}{A(0,0)x(0,0) + B(0,0)y(0,0)} \tag{2.67}$$

The transfer function for other leaf nodes are

$$H(s)|_{2^{n-1}j+2^{n-i-1}} = z(i,j)\, H(s)|_{2^{n-1}j} \tag{2.68}$$

The second-order approximation of the exact transfer function $H_i(s)$ for node *i* of the given distributed *RLC* tree has the form of

$$H_i(s) \approx 1 + m_1^i s + m_2^i s^2 = \frac{1}{s^2 b_2^i + s b_1^i + 1} \tag{2.69}$$

where b_1^i and b_2^i are defined as

$$b_1^i = -m_1^i \tag{2.70}$$

$$b_2^i = \left(m_1^i\right)^2 - m_2^i \tag{2.71}$$

For a step input, the time domain response at node *i* can be obtained using (2.34) – (2.37), from which the 10%, 50% and 90% delays characterizing the time domain response can be estimated. The two-pole model [25] provides a method, which can simulate the distributed *RLC* tree structure fast and accurately by using a second order Maclaurin series to approximate the cosh and sinh terms of (2.7). The two-pole model [25] provides more accurate moments for transfer function (2.69) than the moments obtained from the lumped *RLC* model due to the fact that the two-pole model [25] for distributed *RLC* tree structure is derived directly from the exact hyperbolic functions (2.10) – (2.13). However, the two-pole model [25] cannot be guaranteed always stable mathematically. In (2.69), the coefficient b_1 is independent of interconnect inductance, coefficient b_2 is the function of interconnect inductance. In some cases, large value of interconnect inductance leads to a minus b_2, which causes the second order approximation of moment matching instable. Theoretically, any needed order approximation of the exact transfer functions for distributed *RLC* interconnect tree can be obtained with desired accuracy according to the scheme of [25]. However, increasing the number of poles by using a higher order Maclaurin series to approximate the cosh and sinh terms of (2.10) and (2.13) following the steps of [25] results in unstable three-pole models, which will be discussed in numerical example Section.

2.4.2.2 Implicit Moment Matching

Implicit moment matching technique uses Krylov subspace to project large matrices on its dominant eigenspace [56], [57]. Taking PRIMA (Passive Reduced-Order Interconnect Macromodeling Algorithm) as an example to explain the basic concept of implicit moment matching technique.

PRIMA is a Krylov subspace based projection method which generates guaranteed stable and provably passive reduced order *N*-port models for *RLC* interconnect circuits [29]. Briefly, PRIMA is an orthogonal projection method which takes a linear circuit in the form

$$\boldsymbol{G}\boldsymbol{x} + \boldsymbol{C}\frac{d\boldsymbol{x}}{dt} = \boldsymbol{B}\boldsymbol{u}_p(t) \tag{2.72}$$

$$\boldsymbol{i}_p(t) = \boldsymbol{L}^T\boldsymbol{x} \tag{2.73}$$

The $\boldsymbol{i}_p$ and $\boldsymbol{u}_p$ vectors denote the port currents and voltages, respectively, and

$$\boldsymbol{C} \equiv \begin{bmatrix} \boldsymbol{Q} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{H} \end{bmatrix} \quad \boldsymbol{G} \equiv \begin{bmatrix} \boldsymbol{N} & \boldsymbol{E} \\ -\boldsymbol{E}^T & \boldsymbol{0} \end{bmatrix} \quad \boldsymbol{x} \equiv \begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{i} \end{bmatrix} \tag{2.74}$$

where $\boldsymbol{v}$ and $\boldsymbol{i}$ are the modified nodal analysis (MNA) variables producing a total number of *n* unknows in (2.72) and (2.73), which correspond to the node voltages and the branch currents for voltage sources and inductors, respectively. The matrices $\boldsymbol{G} \in \Re^{n\times n}$ and $\boldsymbol{C} \in \Re^{n\times n}$ represent the conductance and susceptance matrices. $\boldsymbol{N}$, $\boldsymbol{Q}$ and $\boldsymbol{H}$ are the matrices containing the stamps for resistors, capacitors, and inductors, respectively. $\boldsymbol{E}$ consists of ones, minus ones and zeros, which represent the current variables in KCL equations. PRIMA can find a reduced-order model in the form

$$\boldsymbol{V}_q^T\boldsymbol{G}\boldsymbol{V}_q\boldsymbol{x}_q + \boldsymbol{V}_q^T\boldsymbol{C}\boldsymbol{V}_q\frac{d\boldsymbol{x}_q}{dt} = \boldsymbol{V}_q^T\boldsymbol{B}\boldsymbol{u}_p \tag{2.75}$$

$$\boldsymbol{i}_p(t) = \boldsymbol{L}^T\boldsymbol{V}_q\boldsymbol{x}_q \tag{2.76}$$

In order to preserve passivity, PRIMA requires a minor modification in the MNA formulation of the original circuit. Considering a unit impulse voltage at the ports and taking the Laplace transformation of (2.72) and (2.73), there is

$$(s\boldsymbol{C} + \boldsymbol{G})\boldsymbol{X} = \boldsymbol{B} \tag{2.77}$$

Rewriting (2.77) as

$$(\boldsymbol{I} - s\boldsymbol{A})\boldsymbol{X} = \boldsymbol{B} \tag{2.78}$$

where $\boldsymbol{A} = -\boldsymbol{G}^{-1}\boldsymbol{C}$, $\boldsymbol{R} = -\boldsymbol{G}^{-1}\boldsymbol{B}$ and $\boldsymbol{I}$ is the identity matrix.

Having obtaining the required circuit formulation, the next step is to find the Krylov subspace to be used in the projection, which is important for the accuracy and size of the reduced-order system. The simplest way to obtain the transformation matrix $\boldsymbol{V}_k$ is to use a block Arnoldi algorithm with an expansion around s = 0. Define the moment matrix as

$$K = \left[\boldsymbol{M}_0\ \boldsymbol{M}_1\ \boldsymbol{M}_2\ \ldots\ \boldsymbol{M}_{q-1}\right] \ = [\boldsymbol{R}\ \ \boldsymbol{AR}\ \ \boldsymbol{A}^2\boldsymbol{R} \ldots\ \boldsymbol{A}^{q-1}\boldsymbol{R}] \tag{2.79}$$

where q is the number of moments and is determine by the approximation order k and port numbers p,

$$q = \lfloor k/p \rfloor \tag{2.80}$$

By using a numerically well conditioned algorithm known as the block Arnoldi algorithm to construct the transformation matrix $\boldsymbol{V}_k$ in the following manner,

Finding $\boldsymbol{V}_0$

$$\boldsymbol{V}_0 = \frac{\boldsymbol{M}_0}{\|\boldsymbol{M}_0\|} \tag{2.81}$$

Finding $\boldsymbol{V}_1$

$$\boldsymbol{q}_1 = \boldsymbol{A}\boldsymbol{V}_0$$

$$\boldsymbol{q}_1' = \boldsymbol{q}_1 - (\boldsymbol{q}_1\boldsymbol{V}_0)\boldsymbol{V}_0$$

$$\boldsymbol{V}_1 = \frac{\boldsymbol{q}_1'}{\|\boldsymbol{q}_1'\|} \tag{2.82}$$

Finding $\boldsymbol{V}_2$

$$\boldsymbol{q}_2 = \boldsymbol{A}\boldsymbol{V}_1$$

$$\boldsymbol{q}_2' = \boldsymbol{q}_2 - (\boldsymbol{q}_2 \boldsymbol{V}_0)\boldsymbol{V}_0 - (\boldsymbol{q}_2 \boldsymbol{V}_1)\boldsymbol{V}_1$$

$$\boldsymbol{V}_2 = \frac{\boldsymbol{q}_2'}{\|\boldsymbol{q}_2'\|} \tag{2.83}$$

Continue this process until the *k*th moment is obtained. The transformation matrix $\boldsymbol{V}_k$ is formed as

$$\boldsymbol{V}_k = [\boldsymbol{V}_0 \ \boldsymbol{V}_1 \ \boldsymbol{V}_2 \ \ldots \ \boldsymbol{V}_{k-1} \ ] \tag{2.84}$$

After obtaining the transformation matrix $\boldsymbol{V}_k$, the reduced-order model is constructed as

$$\boldsymbol{G}_k \boldsymbol{x}_k + \boldsymbol{C}_k \frac{d\boldsymbol{x}_k}{dt} = \boldsymbol{B}_k \boldsymbol{u}_p \tag{2.85}$$

$$\boldsymbol{i}_p(t) = \boldsymbol{L}_k^T \boldsymbol{x}_k \tag{2.86}$$

where the reduced-order system matrices can be obtained using the congruence transformations,

$$\boldsymbol{C}_k = \boldsymbol{V}_k^T \boldsymbol{C} \boldsymbol{V}_k$$

$$\boldsymbol{G}_k = \boldsymbol{V}_k^T \boldsymbol{G} \boldsymbol{V}_k$$

$$\boldsymbol{B}_k = \boldsymbol{V}_k^T \boldsymbol{B}$$

$$\boldsymbol{L}_k = \boldsymbol{V}_k^T \boldsymbol{L} \tag{2.87}$$

In (2.85) and (2.86), the reduced-order system reduces the size of the original system from *n* unknows to *k* unknows, where *k* is generally much smaller than *n*. In order to

simulate high speed systems, designers need to analyze accurate electromagnetic models of the interconnect and package together with their drivers and receivers in a circuit simulation environment. The *s*-domain models of transmission lines, however, are not compatible with the PRIMA formulation, hence they cannot be used directly. The brute force approach is to model them with multiple *RLC* segments. But in addition to the possible accuracy loss, this approach also increases the size of the system to be reduced. Although the Krylov subspace methods, are, in general, numerically much more robust than explicit moment matching, they still have limitations on the approximation orders. Due to finite machine precision, the ever-generated Krylov vectors eventually lose orthogonality and the method using them stagnates. The effect of this phenomenon on the accuracy is identical to what happens in AWE: including more Krylov vectors does not necessarily increase the quality of the approximation after some order.

But more important, a rank-deficient projection matrix due to orthogonality loss can yield unstable and therefore nonpassive reduced order models. PRIMA preserving passivity holds only under the assumption that reduced-order matrices in (2.87**)** are positive semidefinite. This assumption in turn requires a full-rank projection matrix. Thus, even if the reduced-order model turns out to be stable, the passivity with a projection matrix that does not possess full rank cannot be guaranteed.

2.5 Conclusion

This chapter, in summary, presents an overview of quasi-TEM models and full wave models. All closed-form *RLC* models assume a quasi-TEM mode of signal propagation. In addition, MOR algorithm based explicit and implicit moment matching techniques are discussed. For explicit moment matching techniques using a single Padé expansion, if the rational approximation contains more than eight poles, the single Padé expansion may produce inaccurate results and unstable poles. Another problem with explicit moment matching techniques is that the reduced-order macromodel is not guaranteed to be passive. Conversely, the implicit moment matching techniques can preserve the passivity of the reduced-order system and capture more poles from a single expansion. However, there exists limitation in quality of the approximation and possibility in loss of passivity.

Emphasizing the existing difficulties for current algorithms in the literature and the need for a new method, which satisfies the accuracy and efficiency is the subject of the following chapters.

Chapter 3

Development of the Proposed Analytic Delay Model

3.1 Introduction

As discussed in previous chapter, the approximation of signal delay provided by explicit moment matching based two-pole model [25] is not accurate enough. Thus, the two-pole model [25] cannot capture the early transient responses required for estimating long signal delays produced by electrically long *RLC* interconnect circuits and inductive dominant on-chip interconnects. In addition, the two-pole model [25] may generate unstable poles and cannot guarantee the passivity of the reduced-order macromodels. PRIMA [29], an efficient implicit moment matching based algorithm, can provide accurate *RLC* reduced order models for any approximation order and preserve the passivity of the reduced system without causing numerical issues. However, the reduced system may contain poles that are not dominant making the macromodel unnecessarily large since Krylov methods capture many poles. For linear distributed networks, it is difficult to evaluate their transient responses due to the infinite number of poles.

Hence, it is essential to propose an efficient solution to the above problems. This chapter provides an analytic delay model of *RLC* distributed interconnect using Numerical Inversion of the Laplace Transform (NILT), which is easily applicable for any kind of transfer function, and does not require the knowledge of the poles of the function under consideration. The proposed method can be used for the evaluation of signal delays of various *RLC* distributed interconnect networks. This chapter is organized as follows. Section 3.2 compares the classic inverse Laplace transform with the numerical inversion of Laplace transform in terms of both basic formulae and characteristics. In Section 3.3, the proposed method is developed by introducing Padé rational approximation as its core concept. Section 3.4 summarizes the important properties for the proposed method. In the Section 3.5 and 3.6, the new applications of the proposed method are introduced as well the significant advantages of the proposed method is presented while comparing with other conventional numerically integration methods. The conclusion for this chapter is provided in Section 3.7.

3.2 Classical Laplace Transform vs. the NILT

3.2.1 Comparative Descriptions

According to the descriptions of classical method of inversion, time domain solution of linear networks using the Laplace transform can be accomplished by first finding the appropriate transfer function in the *s*-domain, defining the input signal in terms of the variable *s*, and inverting their product. The inversion involves finding poles and residues. Practical problems do occur: obtaining the transfer function is possible only for relatively small networks, and finding the poles and residues is always fairly expensive. Moreover, accuracy may be impaired if multiple poles occur.

Various numerical Laplace inversion methods which do not require evaluation of the poles are presented in the literature. The proposed method in the work is particularly suitable for network applications. The proposed method does not require determination of the poles and residues. It is applicable to stiff systems, to systems with multiple poles, and to systems with distributed parameter. It can be modified so that it is equivalent to an absolutely stable, very-high-order integration method.

Classical Laplace transform has an additional advantage: it can correctly handle discontinuous functions, like the unit step, or even the Dirac impulses. Classical Laplace transforms can also correctly handle inconsistent initial conditions, as may appear in networks with ideal switches. It turns out that the proposed method retains this advantageous property and can be used as an integration method across the switching instants.

3.2.2 Basis of the Proposed Method

As discussed in previous section, the evaluation of time responses is usually based on the formulae for partial fraction expansion when using classical inverse Laplace transform, for which the enough knowledge of the poles of the function under consideration is needed. If the function is a rational one and the poles of the function are finite, the inversion into the time domain can be obtained by using expansion into partial fractions and the tables of pairs of the Laplace transform.

For the network containing distributed elements, the Laplace complex operator *s* may appear under the square root sign. For lossless distribute lines, the Laplace complex operator *s* is introduced in the form of e^{sk} , whereas the form $e^{\sqrt{sk}}$ is introduced for distributed *RC* lines. The parameter *k* here is a number, which expresses some combination of elements per-unit-length. If only s or $\sqrt{s}$ is presented, the exact solution is easily to obtain by transforming $\sqrt{s} = z$ and decomposing $F(z)$ into partial fractions in terms of z. For simple poles [58],

$$\mathcal{L}^{-1}\big(F(z)\big) = \mathcal{L}^{-1}\left(\sum \frac{A_i}{z - a_i}\right) = \mathcal{L}^{-1}\left(\sum \frac{A_i}{\sqrt{s} - a_i}\right) \tag{3.1}$$

and the inverse can be found by looking up the tables.

However, if e^{sk} or $e^{\sqrt{sk}}$ are presented, the situation becomes much more difficult. In some simple cases, exact solutions are still possible [59]. For more complicated cases, some types of approximations are required. Typically, two kinds of approximations are applied. The first method, which is applicable for monotonic transient responses to a unit step, requires all of the infinite number of poles to lie on the negative real axis and evaluates special coefficients to obtain the delay and the rise time of the responses. The numerator and denominator of the transfer function are expanded in Taylor series as,

$$F(s) = \frac{1 + sa_1 + s^2a_2 + \cdots + s^n a_n}{1 + sb_1 + s^2b_2 + \cdots + s^m b_m} \tag{3.2}$$

The delay, which is the time for the response to reach half its final value, is defined as

$$T_D = b_1 - a_1 \tag{3.3}$$

In [24], variables b_1 and a_1 are defined as

$$b_1 = \sum_{i=1}^{m} \frac{1}{p_i} \tag{3.4}$$

and

$$a_1 = \sum_{i=1}^{n} \frac{1}{z_i} \tag{3.5}$$

where p_i and z_i are the poles and zeros of the transfer function, respectively. Hence, the delay of (3.3) is treated as the reciprocal of the dominant pole of the system. The approximation given by (3.3) is accurate for systems that can be modeled by a single dominant pole and has no low-frequency zeros near the dominant pole. Using the approximation, the step response of the system can be expressed as

$$v(t) = 1 - e^{\left(-\frac{t}{T_D}\right)} \tag{3.6}$$

Similarly, the rise time is defined as

$$T_R = \sqrt{\left(2\pi\left(b_1^2 - a_1^2 + 2(a_2 - b_2)\right)\right)} \tag{3.7}$$

Observed from (3.7), the rise time T_R is the reciprocal of the tangent at this point.

The second method, a more general method, was provided in [60]. The application of this method requires to expand the transfer function of the distributed networks similarly as (3.2) and to truncate the expansions after some terms. Thus, the infinite number of the poles is replaced by a finite distribution of poles and the distributed two-port is substituted by its lumped approximation. However, the main difficulty for this method is due to the fact that it needs a considerable number of steps before evaluating the approximate response: expanding the function or functions, evaluating the poles (usually the most difficult problem), evaluating the residues and performing the inverse transformation. The method is not very practical since all the mentioned steps must be performed for each element connection.

The proposed method in this chapter is similar to the second method mentioned above, however, the proposed method avoids all the expansions, root finding and partial fraction expansions. For any function, the problem is reduced to insertion of fixed complex numbers instead of the variable *s*.

3.3 Development of the Proposed Method

The proposed algorithm is based on the numerical inversion of the Laplace transform (NILT) which was previously used as an integration formula for SPICE analysis of interconnect circuits [43]-[47]. NILT directly uses the frequency solution of interconnect circuits without using macromodeling algorithms (i.e. such as lumped *RLC* circuits [47]) or moment matching techniques [26]-[32] to approximate the transfer function of interconnects. Since NILT is equivalent to an integration formula that is stable for higher order approximations [43], [45], the proposed algorithm develops an analytic delay model for on-chip *RLC* interconnects, and provides a mechanism to increase the accuracy of the model for electrically long *RLC* interconnect circuits. Furthermore, the higher order time functions obtained by NILT can be solved in parallel using multi-core processing techniques.

3.3.1 Formulae for the Proposed Method

The development of the proposed method is based on the inverse Laplace transform formula [45], [58],

$$v(t) = \frac{1}{2\pi j}\int_{c-j\infty}^{c+j\infty} V(s)e^{st}ds \tag{3.8}$$

where s is the Laplace complex operator, c is an arbitrary positive constant such that $Re\ s_i < c$, where s_i are the poles of $V(s)$, and $V(s)$ is generally a function of s, $\sqrt{s}$, e^{sk} and $e^{\sqrt{sk}}$. The exact inversion as (3.8) is possible only if the poles of $V(s)$ are known. As discussed in Section 3.2, the distributed network has an infinite number of poles, it is necessary to avoid the complicated root finding procedure by applying some approximation to the integrand in (3.8). The approximation could be done on $V(s)$, however, the approximation has to be done all over again for each new problem. Instead of approximating $V(s)$, it is equally justifiable to substitute e^{st} of the integrand in (3.8) by some convenient approximation, which has significant advantage since such an approximation can be found conclusively and need to be evaluated only once. Such an approximation to e^{st} constructs the core concept of the proposed method.

3.3.2 Simple Case of Approximation Order

There are many kinds of approximation for e^{st}, one of which is the rational approximation. When the numerator and denominator have the same degree, the rational approximation has a special advantage, that is, all the poles are located in the right half-plane for any degree of approximation, all of them are simple, and the zeros of the function are mirror images of the poles about the imaginary axis. For simplicity, suppose the degrees of the numerator and denominator are the same and approximation of e^{st} is expressed as [58]

$$e^{st} \approx \xi_k(st) = \frac{E_k(st)}{E_k(-st)} = \frac{\sum_{i=0}^{k} C_i^{(k)}(st)^i}{\sum_{i=0}^{k} C_i^{(k)}(-st)^i} \tag{3.9}$$

where

$$C_i^{(k)} = \frac{(2k-i)!}{(k-i)!i!} \quad i = 0, 1, 2, \dots, k \tag{3.10}$$

Suppose that $V(s)$ is stable and that it does not contain any entire function. For an arbitrary function, the stability requirement does not imply that all poles are lying in the interior of the left half plane, however, such an assumption is nevertheless acceptable for functions used to describe realizable networks in [61]. All of the poles of $\xi_k(st)$ are located in the right half plane and are separated from the poles of $V(s)$ by a line parallel to the imaginary axis. Though, for the sake of simplicity, it is assumed that all the poles of $\xi_k(st)$ are in right half plane due to the same degrees of the numerator and denominator of the rational approximation to e^{st}, the left half plane poles of (3.9) do not alter the following results.

Inserting (3.9) into (3.8), the approximation $\hat{v}(t)$ of $v(t)$ becomes

$$\hat{v}(t) = \frac{1}{2\pi \mathrm{j}} \int_{c'-j\infty}^{c'+j\infty} V(s)\, \xi_k(st) ds \tag{3.11}$$

(3.11) can be evaluated as a sum of the residues in one half plane only. In the selected right half plane, the poles of approximation $\xi_k(st)$ and their residues are known. Define a new variable *z* and introduce the following transformation as

$$st = z \tag{3.12}$$

The alternate in (3.12) is used to remove the variable *t* from e^{st} and then to approximate e^{st}. This is done numerically only once in the development of the proposed method, and all further inversions using the results are thus obtained.

Inserting (3.12) into (3.11), the approximation $\hat{v}(t)$ in (3.11) can be rewritten as

$$\hat{v}(t) = \frac{1}{2\pi \mathrm{j} t} \int_{c'-j\infty}^{c'+j\infty} V\left(\frac{z}{t}\right) \xi_k(z) dz \tag{3.13}$$

For (3.9), the partial fraction expansion is

$$\xi_k(z) = (-1)^k + \sum_{i=1}^{k} \frac{\alpha_i + j\beta_i}{z - (a_i + jb_i)} \tag{3.14}$$

Inserting (3.14) into (3.13) leads to

$$\hat{v}(t) = \frac{1}{2\pi \mathrm{j} t} \int_{c'-j\infty}^{c'+j\infty} (-1)^k V\left(\frac{z}{t}\right) dz + \frac{1}{2\pi \mathrm{j} t} \int_{c'-j\infty}^{c'+j\infty} \sum_{i=1}^{k} (\alpha_i + j\beta_i) \frac{V\left(\frac{z}{t}\right)}{z - (a_i + jb_i)} \, dz \tag{3.15}$$

Since it is assumed that $V(s)$ has no poles located in the right half plane, the first integral in (3.15) is equal to zero. The second integral of (3.15) has only simple poles and equals

$$\hat{v}(t) = \frac{1}{t} \sum_{i=1}^{k} (\alpha_i + j\beta_i) \, V\left(\frac{a_i + jb_i}{t}\right) \tag{3.16}$$

In (3.16), $(\alpha_i + j\beta_i)$ are the residues and $(a_i + jb_i)$ are the poles of the approximation (see Table 3.1). Equation (3.16) is actually the final time domain formula for a general frequency function $V(s)$. For any time variable *t*, the coordinates of the poles are divided by *t*, inserted instead of z/t into $V(z/t)$ and the sum (3.16) is evaluated. One of the important features of (3.16) is that it is not restricted to equal time increases. It is applicable for evaluating the step responses of distributed element networks. To be able to make some practical criteria for the amount of the error possible, which are generated from the use of a given degree of the approximation $\xi_k(z)$, exact solutions are necessary for comparison.

Table 3. 1: Poles and residues of the approximation of e^z

Approximation Order k	Poles $p_i = a_i + jb_i$		Residues of p_i	
	a_i	b_i	α_i	β_i
2	3.0	1.7320508	-6.0	10.392305
3	4.6443707	0.0	57.202540	0.0
	3.6778146	3.5087619	-16.601270	-20.583184
4	5.7924212	1.7344683	-66.319948	173.25872
	4.2075788	5.3148361	46.319948	-16.890455
5	7.2934772	0.0	810.97975	0.0
	6.7039128	3.4853228	-378.10448	-303.11741
	4.6493486	7.1420458	2.6146049	82.787193
6	8.4967188	1.7350193	-819.75569	2612.0908
	7.471416	5.2525446	902.09509	-626.70118
	5.0318645	8.9853459	-124.33940	-36.416754
7	9.9435738	0.0	11489.262	0.0
	9.516581	3.4785721	-6522.8514	-4274.0234
	8.1402783	7.0343481	725.08042	2130.1726
	5.3713538	10.841388	109.13998	-158.40896
8	11.175772	1.735229	-10643.848	38247.170
	10.409682	5.2323502	14672.653	-13083.555
	8.7365784	8.828885	-4265.9984	219.87981
	5.6779679	12.707823	165.19301	219.15799
9	12.594039	0.0	162754.94	0.0
	12.258735	3.4756969	-103155.57	-59868.288
	11.208844	6.996314	20518.457	40227.301
	9.2768797	10.634543	1709.6717	-7445.9948
	5.9585216	14.582927	-360.02803	119.56047
10	13.84409	1.7353300	-141902.30	553065.81
	13.230582	5.2231360	224511.30	-230851.63
	11.935056	8.7698940	-94119.473	20760.230

	9.7724400	12.449970	11396.208	6301.1275
	6.217832	16.465338	4.2664130	-511.07576
11	15.244680	0.0	2305564.7	0.0
	14.968460	3.474206	-1568357.0	-838476.68
	14.115784	6.97803	422867.39	677638.94
	12.602674	10.552384	7319.7044	-193630.93
	10.231296	14.274042	-15115.640	15053.525
	6.459444	18.354224	635.18259	225.41453
12	16.506844	1.735388	-1922168.6	7945487.9
	15.994542	5.218134	3340003.4	-3773342.2
	14.931142	8.74034	-1756038.5	577890.96
	13.222008	12.34307	354728.44	107516.64
	10.659418	16.105814	-16133.768	-29701.897
	6.686046	20.248594	-546.91141	680.36041
13	17.89542	0.0	32661018.0	0.0
	17.660504	3.473332	-23341901.0	-11757148.0
	16.941184	6.967736	7631837.8	10787865.0
	15.68876	10.509806	-313414.86	-4025999.2
	13.800746	14.141288	-357188.70	578423.44
	11.061362	17.944496	50925.875	-10766.856
	6.899734	22.147858	-586.03207	-946.45309
14	19.166342	1.735422	-26308359.0	113711702.0
	18.726292	5.215106	48937813.0	-59144660.0
	17.822002	8.723208	-30084712.0	12233846.0
	16.397694	12.286082	8285118.1	1369919.0
	14.344792	15.946434	-824846.10	-875087.97
	11.440704	19.789416	-6593.2088	77927.299
	7.102174	24.051476	1369.2445	-294.84208

These can be best obtained in the case of rational functions. The method can be further simplified for the application for lumped network functions. In order to simplify the rational function, consider the rational function below

$$V(s) = \frac{P_m(s)}{Q_n(s)} = \frac{\sum_{r=1}^{m+1} d_r s^{r-1}}{\sum_{r=1}^{n+1} g_r s^{r-1}} \qquad m < n \tag{3.17}$$

and

$$v(t) = \frac{1}{t} \sum_{i=1}^{k} (\alpha_i + j\beta_i) \frac{P_m\left(\frac{a_i + jb_i}{t}\right)}{Q_n\left(\frac{a_i + jb_i}{t}\right)} \tag{3.18}$$

The insertion of (3.17) into (3.18) leads to

$$v(t) = t^{n-m-1} \sum_{i=1}^{k} (\alpha_i + j\beta_i) \frac{\sum_{r=1}^{m+1} d_r (a_i + jb_i)^{r-1} t^{m+1-r}}{\sum_{r=1}^{m+1} g_r (a_i + jb_i)^{r-1} t^{n+1-r}} \tag{3.19}$$

By using the following recurrent formula, the powers of the pole coordinates can be numerically determined. Set for the zeroth power

$$E_{1,i} = 1$$

$$F_{1,i} = 0 \tag{3.20}$$

and perform the following multiplication

$$\left(E_{1,i} + jF_{1,i}\right)(a_i + jb_i) = E_{1,i}a_i - F_{1,i}b_i + j\left(F_{1,i}a_i + E_{1,i}b_i\right) = E_{2,i} + jF_{2,i} \tag{3.21}$$

Repeating the multiplication application of (3.21) results in

$$E_{r+1,i} = E_{r,i}a_i - F_{r,i}b_i$$

$$F_{r+1,i} = F_{r,i}a_i + E_{r,i}b_i \tag{3.22}$$

After knowing the powers of the poles, (3.19) can be simplified to

$$v(t) = t^{n-m-1} \sum_{i=1}^{k} (\alpha_i + j\beta_i) \frac{A(t) + jB(t)}{C(t) + jD(t)} \tag{3.23}$$

where

$$A(t) = \sum_{r=1}^{m+1} d_r E_{r,i}\, t^{m+1-r}$$

$$B(t) = \sum_{r=1}^{m+1} d_r F_{r,i}\, t^{m+1-r}$$

$$C(t) = \sum_{r=1}^{m+1} g_r E_{r,i}\, t^{n+1-r}$$

$$D(t) = \sum_{r=1}^{m+1} g_r F_{r,i}\, t^{n+1-r} \tag{3.24}$$

3.3.3 General Case of Approximation Order

For the sake of clarity, rewrite here the Laplace transform inversion formula,

$$v(t) = \frac{1}{2\pi j} \int_{c-j\infty}^{c+j\infty} V(s) e^{st} ds \tag{3.25}$$

Using the transformation introduced in previous section,

$$st = z \tag{3.26}$$

Substituting (3.26) into (3.25), time domain response can be expressed as

$$v(t) = \frac{1}{2\pi j} \int_{c'-j\infty}^{c'+j\infty} V\left(\frac{z}{t}\right) e^{z} dz \tag{3.27}$$

Using next a rational function to approximate the function e^z in (3.27), where the Padé approximation is used

$$e^z \approx \xi_{N,M}(z) = \frac{P_N(z)}{Q_M(z)} = \frac{\sum_{i=1}^{N} \frac{(M+N-i)!N!}{(M+N)!i!(N-i)!} z^i}{\sum_{i=1}^{M} \frac{(M+N-i)!M!}{(M+N)!i!(M-i)!} (-z)^i} \tag{3.28}$$

where $P_N(z)$ and $Q_M(z)$ are polynomials of order *N*, *M*, respectively. It is well known that the Padé approximation formally equates a rational function to some terms of a series

$$\frac{\sum_{i=0}^{N} a_i z^i}{1 + \sum_{i=0}^{M} b_i z^i} = \sum_{i=0}^{M+N} C_i z^i + \sum_{i=M+N+1}^{\infty} C_i z^i \tag{3.29}$$

and in this case the coefficient C_i for $i \leq M + N$ are the coefficients of the Taylor expansion of e^z. As a result, the Padé approximation $\xi_{N,M}(z)$ has the first $M + N + 1$ terms of its Taylor expansion of e^z. However, the remining terms of both Padé rational approximation and Taylor expansion differ.

It is not necessary to solve the system of equations arising from (3.29) since a closed form exists [62],

$$\xi_{N,M}(z) = \frac{P_N(z)}{Q_M(z)} = \frac{\begin{array}{r}(M+N)! + (M+N-1)!\binom{N}{1} z + (M+N-2)!\binom{N}{2} z^2 \\ + \cdots + M!\binom{N}{N} z^N\end{array}}{\begin{array}{r}(M+N)! - (M+N-1)!\binom{N}{1} z + (M+N-2)!\binom{N}{2} z^2 \\ + \cdots + (-1)^M N!\binom{N}{N} z^N\end{array}} \tag{3.30}$$

Several of the first approximations are shown in Table 3.2.

As $\lim(M + N) \to \infty$, any sequence of these functions in Table 3.2 converges to e^z for any *z*, which has been proved in [62]. The poles of all approximations in (3.30) are simple, for *M* not differing considerably from *N*, all are in the right half plane. This fact is required in the following detailed explanation of the proposed method. Inserting (3.30) into (3.27), $v(t)$ can be approximated as $\hat{v}(t)$,

$$\hat{v}(t) = \frac{1}{2\pi \mathrm{j} t} \int_{c'-j\infty}^{c'+j\infty} V\left(\frac{z}{t}\right) \xi_{N,M}(z) dz \tag{3.31}$$

Table 3. 2: Padé table for the approximation of e^z

N / M	0	1	2
0	$\frac{1}{1}$	$\frac{1+z}{1}$	$\frac{\frac{z^2}{2}+z+1}{1}$
1	$\frac{1}{z}$	$\frac{1+\frac{z}{2}}{1-\frac{z}{2}}$	$\frac{\frac{z^2}{6}+\frac{2z}{3}+1}{1-\frac{z}{3}}$
2	$\frac{1}{\frac{z^2}{2}-z+1}$	$\frac{1+\frac{z}{3}}{\frac{z^2}{6}-\frac{2z}{3}+1}$	$\frac{\frac{z^2}{12}+\frac{z}{2}+1}{\frac{z^2}{12}-\frac{z}{2}+1}$

Applying residue calculus to (3.31), the integral here can be evaluated by closing the path of integration along an infinite arc either to the right or to the left. In order to ensure that the path along the infinite arc does not contribute to the integral of (3.31), choose *M*, *N* such that the function

$$F(z) = V\left(\frac{z}{t}\right)\xi_{N,M}(z) \tag{3.32}$$

has at least two more finite poles than zeros. Then function $F(z)$ fulfills

$$\oint F(z)dz = \pm 2\pi j \sum (\text{residues at poles inside the closed path}) \tag{3.33}$$

where the negative sign applies when the path is closed in the right half plane (clockwise), whereas the positive one applies for the other case. For *N* < *M*,

$$\xi_{N,M}(z) = \sum_{i=1}^{M} \frac{K_i}{z - z_i} \tag{3.34}$$

where z_i are the poles of approximation $\xi_{N,M}(z)$, and K_i are the corresponding residues.

Closing the path of integration around the poles of $\xi_{N,M}(z)$ in the right half plane [43]

$$\hat{v}(t) = -\frac{1}{t}\sum_{i=1}^{M} K_i\, V\left(\frac{z_i}{t}\right) \tag{3.35}$$

This is the basic inversion formula. Real-time functions can be evaluated using only the poles z_i in the upper half plane. This reduces the computations to one half. For M even and bar denoting a complex conjugate,

$$\hat{v}(t) = -\frac{1}{t}\sum_{i=1}^{M'} K_i\, V\left(\frac{z_i}{t}\right) - \frac{1}{t}\sum_{i=1}^{M'} \overline{K}_i\, V\left(\frac{z_i}{t}\right)$$

$$= -\frac{1}{t}\sum_{i=1}^{M'} 2\mathrm{Re}\left(K_i V\left(\frac{z_i}{t}\right)\right)$$

$$= -\frac{1}{t}\sum_{i=1}^{M'} \mathrm{Re}\left(K_i' V\left(\frac{z_i}{t}\right)\right) \tag{3.36}$$

where $M' = \frac{M}{2}$ and $K_i' = 2K_i$, K_i is defined by (3.34). When M is odd, $M' = \frac{M+1}{2}$ and $K_i' = K_i$ for the residue corresponding to the real pole. Since calculations must be done in complex, nothing is gained by using odd M. In the following work, M is assumed to be even. Due to the application of (3.36), the computational effort is essentially halved.

The pole z_i and residues K_i' can be calculated with high precision, and a selection is given in Table 3.3. A more detailed table is given in [43]. For the distributed networks, their elements are described by partial differential equations, and direct application of the numerical integration methods, such as the forward Euler formula, the backward Euler formula and the trapezoidal rule, are not possible. Simple distributed elements can be described at their terminals through a chain matrix representation involving transcendental function of *s*. In very simple cases, the time domain method can be obtained as an infinite series. The numerical Laplace transform method is applicable whenever the terminal description is available in the *s*-domain.

In numerical examples section of the following chapter, select $N = M - 1$, where $M = 2, 4, 8, 16$. Then the pole z_i of $\xi_{N,M}(z)$ and their corresponding residues K_i' can be calculated, and the results are presented in Table 3.4. It is shown in [43] that (3.36) is equivalent a time integration operation with the order of the integration equal to $M + N$. The Padé rational approximation of (3.34), for $N = M - 1$, produces integration formulas that are always stable [43]. In addition to the ability in providing an analytic formula to calculate the time domain response, the summation terms in expression of (3.36) are also independent of each other, thus these summation terms can be solved in parallel using multi-core processors to obtain the time domain response.

3.4 Computational Complexity

From a circuit simulator point of view, the time domain response of (3.36) requires solving the transfer function of (2.11) or (2.14) for a specific pole z_i and time point $t = t_x$. The computational complexity for determining and solving the transfer function is equivalent to solve the circuit. In order to explain computational complexity clearly, concept of Modified Nodal Analysis (MNA) is employed here. For an arbitrary distributed interconnect network, the frequency domain MNA can be written as [44]

$$\left(s\boldsymbol{C} + \boldsymbol{G} + \sum_{k=1}^{N_s} \boldsymbol{D}_k \boldsymbol{A}_k \boldsymbol{D}_k^t \right) \boldsymbol{X} = \boldsymbol{E}(s) \tag{3.37}$$

Then the output voltages of interest can be solved as

$$V_{N_i} = \boldsymbol{\Phi X} = \boldsymbol{\Phi} \left(s\boldsymbol{C} + \boldsymbol{G} + \sum_{k=1}^{N_s} \boldsymbol{D}_k \boldsymbol{A}_k \boldsymbol{D}_k^t \right)^{-1} \boldsymbol{E}(s) \tag{3.38}$$

where $\boldsymbol{C} \in \Re^{N_\pi \times N_\pi}$ and $\boldsymbol{G} \in \Re^{N_\pi \times N_\pi}$ are constant matrices with entries determined by the lumped linear components, $\boldsymbol{X} \in \Re^{N_\pi}$ is the vector of node voltage waveforms in frequency domain appended by independent voltage source current and inductor current waveforms, $\boldsymbol{E}(s) \in \Re^{N_\pi}$ is the vector of source waveforms, $\boldsymbol{A}_k$ is the MNA matrix of the *k*th transmission line, which is computed from transmission line parameters, including line

length, the inductance per unit length, the resistance per unit length, the capacitance per unit length and the conductance per unit length, $\boldsymbol{D}_k = [D_{i,j}]$ with $D_{i,j} \in \{0,1\}$, $i \in \{1, 2, \dots, N_\pi\}$ and $j \in \{1, 2, \dots, 2m_k\}$, is an index matrix with a maximum of one nonzero in each row or column, where m_k is the number of conductors of the *k*th coupled transmission line, N_s is the number of transmission lines, N_π is the dimension of the network equations, $\boldsymbol{\Phi}$ is a selector vector that selects the output voltages of interest.

The MNA formulation of (3.37) can be used to solve the transfer function of V_{N_i} which (3.36) requires to obtain the time domain response. For an arbitrary interconnect network expressed in Norton form, the MNA formulation of (3.37) requires inverting a *w* by *w* matrix to obtain the transfer function of V_{N_i}, where *w* is the number of the nodes contained in the given interconnect network. For the same interconnect network expressed in Thevenin form, the MNA formulation of (3.37) requires inverting a *w*+2 by *w*+2 matrix to obtain the transfer function of V_{N_i}. As a result, the MNA formulation of (3.37) describing the distributed interconnect network in Figure 2.2 requires inverting 4 by 4 matrix to obtain the transfer function of V_{N_1}. For the same network presenting using Norton form, the MNA formulation of (3.37) requires 2 by 2 matrix inversion for the transfer function of V_{N_1}. Similarly, for the interconnect tree structure shown in Figure 2.3, 10 by 10 matrix inversion is required by the MNA formulation of (3.37) for the transfer function of node N_i $(i = 0,1, \cdots 7)$. For its Norton circuit expression, only 8 by 8 matrix inversion is required when using the MNA formulation of (3.37) to obtain the transfer function of node N_i $(i = 0,1, \cdots 7)$. The change of the matrix size when transferring the interconnect network from Norton form to Thevenin form is due to the introduction of an additional node and the independent voltage source current. Thus the main computational complexity requires inverting the MNA equations of (3.37) $M'N_t$ times, where M' is defined by the Padé approximation order M in (3.36) and N_t is the number of time points the transfer function is evaluated. In summary, the computational complexity of the proposed method in this work is mainly determined by the Padé approximation order once the time points number is selected.

In comparison to SPICE based analysis, the distributed interconnect equations of

(3.37) and (3.38) are replaced with the lumped RLC circuits. In SPICE based analysis, the distributed interconnect is modelled by many numbers of lumped *RLC* sections, which greatly increases the size of the MNA matrices. Accordingly, the size of matrix inversion used for calculating the transfer function of node N_i in the given interconnect network is significantly increased. Typically one lumped section adds three variables to the MNA equations (i.e. two node variables and one current variable due to the inductor). The computational complexity for solving the lumped model, requires inverting the MNA equations N_t times. However, the size of the lumped MNA matrices are usually significantly larger for the lumped model when compared to the distributed MNA matrices in (3.37) which directly uses the frequency domain solution of Telegrapher's equation.

In this work, the computational complexity to determine the time domain response of (3.36) for tree circuits at node N_i can be expressed as

$$\Theta\,(M', n_{tr}, n_i) = M' \cdot N_t\big(O(n_{tr}) + O(n_i)\big) \qquad (3.39)$$

where $O(n_i)$ is the computational complexity of computing the transfer function of (2.14) at node N_i for a specific pole z_i and time point $t = t_x$, and n_i is the number of branches along the path from N_0 to N_i. $O(n_{tr})$ is the computational complexity of computing the input impedance at the nodes in the tree and n_{tr} is the number of branches in the entire tree. The input impedances at time point $t = t_x$ and pole z_i are calculated only once for a specific tree. Thus, the additional computational cost to determine the response at another node at a certain time point $t = t_x$ is $M' \cdot O(n_i)$.

For the case when multiple core processors are available, the terms in (3.36) at each frequency pole z_i and time point t_x can be evaluated in parallel. Thus, the computational complexity of (3.39) reduces to

$$\Theta_{\,cp}(M', n_{tr}, n_i) = \left\lceil \frac{M' \cdot N_t}{c_p} \right\rceil \big(O(n_{tr}) + O(n_i)\big) \qquad (3.40)$$

where c_p is the number of core processors and $\lceil \cdot \rceil$ is the ceil operator. The computational complexity analysis discussed here will be illustrated in the following Chapter 4: Numerical Examples.

3.5 Properties of the Proposed Method

The exact inverse Laplace transform of [45]

$$V(s) = \frac{1}{s^m} \tag{3.41}$$

equals

$$v(t) = \frac{t^{m-1}}{(m-1)!} \tag{3.42}$$

for any $m \geq 1$. Insert (3.41) into (3.31),

$$\hat{v}(t) = \frac{t^{m-1}}{2\pi \mathrm{j}} \int_{c'-j\infty}^{c'+j\infty} z^{-m}\, \xi_{N,M}(z) dz \tag{3.43}$$

and integrate by closing the path in the left half plane, where there is only one pole with order of *m* at the origin. Its residue is

$$\mathrm{Res}\left(z^{-m}\, \xi_{N,M}(z) \right)\Big|_{z=0} = \frac{1}{(m-1)!} \frac{d^{m-1}}{dz^{m-1}}\, \xi_{N,M}(z)\Big|_{z=0} \tag{3.44}$$

Since $\xi_{N,M}(z)$ has the first $M+N+1$ terms of its Taylor expansion identical to those of e^z,

$$\frac{d^{m-1}}{dz^{m-1}}\, \xi_{N,M}(z)\Big|_{z=0} = 1 \quad \text{for } m = 1,2,\dots,M+N+1 \tag{3.45}$$

and the inverse is exact

$$\hat{v}(t) = v(t) = \frac{t^{m-1}}{(m-1)!} \quad \text{for } m = 1,2,\dots,M+N+1 \tag{3.46}$$

Insert next (3.34) into (3.43) and integrate by closing the path in the right half plane

Table 3. 3: A selection of pole z_i and residues K_i' for various N, M

M	i	$N = M - 3$		$N = M - 2$		$N = M - 1$	
		$\mathrm{Re}(z_i)$	$\mathrm{Re}(K_i')$	$\mathrm{Re}(z_i)$	$\mathrm{Re}(K_i')$	$\mathrm{Re}(z_i)$	$\mathrm{Re}(K_i')$
		$\mathrm{Im}(z_i)$	$\mathrm{Im}(K_i')$	$\mathrm{Im}(z_i)$	$\mathrm{Im}(K_i')$	$\mathrm{Im}(z_i)$	$\mathrm{Im}(K_i')$
2	1			1.000000000000000	0.000000000000000	2.000000000000000	2.000000000000000
				1.000000000000000	-2.000000000000000	1.414213562373095	-7.071067811865476
4	1	2.764346415715099	-1.48648501159780	3.77901996701019	2.25695874441811	4.78719310312847	26.6030799919430
		1.162323629283279	-12.1091670567457	1.38017652427285	-39.6330870005016	1.56747641689522	-120.143465474095
	2	1.235653584284902	1.48648501159780	2.22098003298981	-2.25695874441814	3.21280689687154	-22.6030799919430
		3.437652493671052	3.43327082695684	4.16039144550693	11.1088316378759	4.77308743327664	24.9433517005007
10	1	10.8209819305222	2186.69723133155	11.8300937391756	16286.6236800444	12.8376770777933	73804.0937628348
		1.51795339370639	-48581.2480583509	1.59375300587506	-139074.711552005	1.66606258418323	-393980.927054819
	2	10.2144903542982	-3989.18174639421	11.2208537793814	-28178.1117127095	12.2261314841909	-122553.999414338
		4.56247943300601	27449.1768403523	4.79296416756808	74357.5823724909	5.01271926367609	190817.197815586
	3	8.93223551465838	2320.95545424058	9.93338372218369	14629.7402524698	10.9343034305873	57833.1445401092
		7.63770336934606	-8164.42299169458	8.03310633426900	-19181.8081849781	8.40967299599511	-36338.3702007367
	4	6.78678737217302	-556.883733516549	7.78114626446220	-2870.41816102606	8.77643464008373	-9310.72169278483
		10.7871525838270	1044.52275257172	11.3688916490480	1674.10948409197	11.9218538983048	3.80354602762962
	5	3.24550482834814	38.4127943302176	4.23452249479722	132.165941247478	5.22545336734471	237.482803799993
		14.1417998906444	-29.0577460615877	14.9570437812819	17.4767479888410	15.7295290456389	282.607384643546

Table 3. 4: A selection of pole z_i and residues K_i' for numerical examples section

i	$N=1, M=2$		$N=3, M=4$		$N=7, M=8$		$N=15, M=16$	
	$\mathrm{Re}(z_i)$	$\mathrm{Re}(K_i')$	$\mathrm{Re}(z_i)$	$\mathrm{Re}(K_i')$	$\mathrm{Re}(z_i)$	$\mathrm{Re}(K_i')$	$\mathrm{Re}(z_i)$	$\mathrm{Re}(K_i')$
	$\mathrm{Im}(z_i)$	$\mathrm{Im}(K_i')$	$\mathrm{Im}(z_i)$	$\mathrm{Im}(K_i')$	$\mathrm{Im}(z_i)$	$\mathrm{Im}(K_i')$	$\mathrm{Im}(z_i)$	$\mathrm{Im}(K_i')$
1	2.000000000000000	2.000000000000000	4.78719310312847	26.6030799919430	10.169446006657910	5225.24169373238	20.8173162340212	209145398.029705
	1.414213562373095	-7.071067811865476	1.56747641689522	-120.143465474095	1.64920179682121	-27098.1202948888	1.69171628049779	-1166830945.51191
2			3.21280689687154	-22.6030799919430	9.40637121369049	-7697.08883087068	20.4322976613954	-420514478.781955
					4.96921728762335	11380.6308676071	5.08129536695889	704018451.418773
3					7.73868814683035	2669.72947491970	19.6460974574755	307479947.156523
					8.37087930623799	-1404.86202099599	8.49034450191054	-234300362.601096
4					4.68549463282126	-189.882337781584	18.4227188359659	-116879277.796634
					12.0105785998138	-50.9533249635550	11.9357249609006	27654339.8083353
5							16.6967416365223	22437451.0172523
							15.4420809018058	5718084.22618171
6							14.3502762952807	-1664686.62959084
							19.0510873566712	-1847999.05506644
7							11.1489235548222	-5830.15187251373
							22.8473895041246	120506.628375914
8							6.48562832451681	1493.35024391385
							27.0674101802434	-466.837429747943

$$\hat{v}(t) = -t^{m-1} \sum_{i=1}^{M} K_i z_i^{-m} \quad \text{for } m = 1,2,\dots,M+N+1 \tag{3.47}$$

Comparing with (3.46) results in

$$-\sum_{i=1}^{M} K_i z_i^{-m} = \frac{1}{(m-1)!} \quad \text{for } m = 1,2,\dots,M+N+1 \tag{3.48}$$

Establishing next a similar rule for positive powers of *s*. The function is now $V(s) = s^m$. Choose *N*, *M* such that (3.32) meets the requirements on the relative number of poles and zeros. Now

$$\hat{v}(t) = \frac{t^{-m-1}}{2\pi j} \int_{c'-j\infty}^{c'+j\infty} z^m \, \xi_{N,M}(z) dz \tag{3.49}$$

The function has no poles to the left of the integrating line. Closing first the path in the left half plane it is concluded that (3.49) is equal to zero. Insert next for $\xi_{N,M}(z)$ from (3.34) close the path in the right half plane with the result

$$\hat{v}(t) = -t^{-m-1} \sum_{i=1}^{M} K_i z_i^m \quad \text{for } 0 \le m \le M-N-2 \tag{3.50}$$

The inequality on the right of (3.50) arises from the condition on the relative number of poles and zeros. Since (3.50) must be fulfilled for any $t > 0$ and it is known already that $\hat{v}(t) = 0$, it follows that

$$\sum_{i=1}^{M} K_i z_i^m = 0 \quad \text{for } 0 \le m \le M-N-2 \tag{3.51}$$

From the derivation above, collect (3.48) and (3.51) into one expression,

$$\sum_{i=1}^{M} \frac{K_i}{z_i^m} = -\frac{1}{(m-1)!} \quad \text{for } -M+N+2 \le m \le M+N+1 \tag{3.52}$$

(3.52) is used to check the numerical accuracy of the poles z_i and residues K_i, and to formulate the following lemma.

Lemma 3.1

Equation (3.34) inverts exactly the function

$$V(s) = s^{-m} \quad \text{for } -M+N+2 \le m \le M+N+1 \tag{3.53}$$

The inverse is

$$\hat{v}(t) = v(t) = \frac{t^{m-1}}{(m-1)!} \tag{3.54}$$

In (3.54), the factorial of a negative number is defined as $\frac{1}{m!} = 0$ for $m < 0$. As a next step in the development of the properties, introduce

$$k = M + N \tag{3.55}$$

Expand $v(t)$ into a Taylor expansion about the origin,

$$v(t) = \sum_{i=0}^{k} \frac{d_i}{i!} t^i + \sum_{i=k+1}^{\infty} \frac{d_i}{i!} t^i \tag{3.56}$$

and consider only the first sum. The Laplace transform of this truncated part is

$$V_T(s) = \sum_{i=0}^{k} \frac{d_i}{s^{i+1}} \tag{3.57}$$

Since each term of $V_T(s)$ is inverted exactly by any $\xi_{N,M}$ fulfilling (3.32), the whole function $V_T(s)$ is also inverted exactly, hence Lemma 3.2 is formulated.

Lemma 3.2

Formula (3.36) inverts exactly the first $M + N + 1$ terms of the Taylor series of any time response. Consequently, as long as the time function is approximated well by the initial

portion of its Taylor series, excellent results will be given by the numerical inversion method. This property indicates that the proposed method is equivalent to the methods for the numerical integration of differential equations. Taking higher *M, N* means that a greater number of the Taylor expansion terms are matched exactly. If a problem is solved twice, once with a given *M*, *N* and the next time with the same *M* but *N* smaller by one, the difference of the two solutions will approximate the (*M* + *N* + l)st term of the Taylor expansion. As long as this term is small, the inversion is likely to be accurate.

Lemma 3.3

In order to establish the validity of the residue calculus, evaluating first the integral

$$I = \int_{c-j\infty}^{c+j\infty} V(s)ds \tag{3.58}$$

with

$$V(s) = \frac{\sum_{i=0}^{N} a_1 s^i}{\sum_{i=0}^{M} b_1 s^i} \tag{3.59}$$

Consider the residue calculus formula

$$I = 2\pi j \times (\text{sum of the residues}) \tag{3.60}$$

In order to apply this formula, close the path by a semicircle in the left half plane and let the radius grow without bound. If the contribution to the integral along this infinite semicircle is zero, the whole integral will be given by the integration along the path indicated in (3.58) and the sum of residues will be the solution.

For the contribution I_1 along the semicircle, introduce a new variable

$$s = Re^{j\emptyset} \tag{3.61}$$

On the semicircle going from *+jR* to *- jR* in the left half plane, the radius *R* is constant, so that

$$ds = jRe^{j\emptyset}d\emptyset \tag{3.62}$$

and

$$I_1 = \int_{+\frac{\pi}{2}}^{-\frac{\pi}{2}} V\left(Re^{j\emptyset}\right) jRe^{j\emptyset}\, d\emptyset \tag{3.63}$$

For every large R the term with the highest powers will dominant, and (3.63) can be simplified as

$$I_1 = \lim_{R\to\infty} \int_{+\frac{\pi}{2}}^{-\frac{\pi}{2}} \frac{a_N R^N e^{j\emptyset N}}{b_M R^M e^{j\emptyset M}}\, jRe^{j\emptyset}\, d\emptyset = \lim_{R\to\infty} \frac{a_N}{b_M} \frac{j}{R^{M-N-1}} \int_{+\frac{\pi}{2}}^{-\frac{\pi}{2}} e^{j\emptyset(N+1-M)}\, d\emptyset \tag{3.64}$$

The last integral on the right is finite. In order to make the whole expression equal to zero, select M and N such that at least the first power of R remains in the denominator. This requires

$$M - N - 1 \geq 1 \tag{3.65}$$

or

$$M \geq N + 2 \tag{3.66}$$

From the analysis above, Lemma 3.3 can be summarized as:

The integral of a rational function $V(s)$ along an infinite semicircle is zero whenever $V(s)$ has at least two more poles than zeros.

When integrating a rational function with $M \geq N + 2$ along the straight line parallel to the imaginary axis, the integral is equal to $2\pi j$ ×(sum of the residues at poles to the left of the line) if the path is closed counterclockwise in the left half plane. If the integration path is closed clockwise in the right half plane, the sum of the residues at poles appearing to the right of the line is taken but is multiplied by $-2\pi j$.

3.6 Comments for Application

Several important comments require to be made although the theoretical results have now been presented.

1. Observed from the formula (3.35), time domain function $v(t)$ cannot be obtained for $t = 0$ using (3.35) on a computer, due to the division by t. To make the situation clarified, consider an example. Let

$$V(s) = \frac{s}{s^2 + a^2} \tag{3.67}$$

Considering the initial value theorem, the corresponding time domain function is equal to $\lim_{t\to 0} v(t) = \lim_{s\to\infty} sV(s) = 1$. Insert next the function above into (3.35). The result becomes

$$-\lim_{t\to 0} \frac{1}{t} \sum_{i=1}^{M} K_i \frac{\frac{z_i}{t}}{\left(\frac{z_i}{t}\right)^2 + (a)^2} = -\sum_{i=1}^{M} K_i \, z_1^{-1} = 1 \tag{3.68}$$

where the last step follows from (3.52). It is shown that the continuous functions $v(t)$ bounded at $t = 0$ are inverted exactly at $t = 0$ by using (3.35) and by the above limiting procedure.

2. Although the inversion in (3.35) cannot be used on a computer for $t = 0$, the proposed method is still suitable for the situation where an impulse appears at the output of a network. In addition, the impulse won't distort the results for $t > 0$, if proper degrees M, N were chosen for the inversion. Since $\mathcal{L}[\delta(t)] = V(s) = 1$, a choice of $N = M - 2$ is sufficient to secure

$$\hat{v}(t) = -\frac{1}{t} \sum_{i=1}^{M} K_i = 0 \quad \text{for } t > 0 \tag{3.69}$$

as follows from (3.52). It should be mentioned at this point that the transfer functions of the network can be realized only if the denominator degree is equal to

or greater than the numerator degree and that the choice of degree satisfying $N = M - 2$ will cover all realizable possibilities. For the situation of a derivative of an impulse nevertheless appearance in the calculations, the choice of $N = M - 3$ will again secure correct answers for $t > 0$. This property is the most important and powerful for the proposed method when applying to the solution of networks having unknown transfer functions. Furthermore, excitations in the form of impulses or their derivatives, present no problems either when applied at $t = 0$. Such excitations usually arise due to inconsistently specified initial conditions on the reactive elements. This is in marked contrast to time domain solutions using differential equations where such excitations cannot be handled simply.

3. The derivatives of time responses are possible using the formula

$$\mathcal{L}\left[v^{(n)}(t)\right] = s^n V(s) - \sum_{i=1} v^{(i-1)}(0+)s^{n-i} \tag{3.70}$$

Choose proper N, M such that $(z/t)^n V(z/t)\,\xi_{N,M}(z)$ fulfills the condition on the relative number of zeros and poles. Then all terms after the Σ operator in (3.70) do not contribute to the inversion and the nth derivative is inverted as

$$\hat{v}^{(n)}(t) = -\frac{1}{t}\sum_{i=1}^{M} K_i \left(\frac{z_i}{t}\right)^n V\left(\frac{z_i}{t}\right) \quad t > 0 \tag{3.71}$$

Furthermore, the integration does not present such difficulties since the number of poles increases.

3.7 Applications of the Proposed Method

The proposed method opens possibilities of novel applications.

1. Such technical quantities as rise time, time delay, overshoot, and undershoot are usually used to define the responses of networks. The numerical method proposed here can solve for these quantities without obtaining the full response. Suppose that $v(t) = \mathcal{L}^{-1}[V(s)]$ is the step response of some network. Then

$$\hat{v}(t) = -\frac{1}{t}\sum_{i=1}^{M} K_i V\left(\frac{z_i}{t}\right) \tag{3.72}$$

and the derivative of $\hat{v}(t)$ is represented by

$$\hat{v}'(t) = -\frac{1}{t}\sum_{i=1}^{M} K_i \left(\frac{z_i}{t}\right) V\left(\frac{z_i}{t}\right) \tag{3.73}$$

as discussed in (3.71). Newton-Raphson iteration can be used to solve for such time t^* for which $\hat{v}(t^*)$ is equal to a desired value *A*:

$$t_{k+1} = t_k - \frac{\hat{v}(t_k) - A}{\hat{v}'(t_k)} = t_k - \left(\frac{\left(\sum_{i=1}^{M} K_i V\left(\frac{z_i}{t_k}\right) + A t_k\right)}{\sum_{i=1}^{M} K_i \frac{z_i}{t_k} V\left(\frac{z_i}{t_k}\right)}\right) \tag{3.74}$$

Almost no any additional computations are required to obtain the derivative, since the evaluation of $V(z_i/t_k)$ will consume most of the time. Similarly, the time to the overshoot or undershoot can be calculated, for which the derivative is equal to zero. The Newton-Raphson formula will then use the first and second derivatives.

2. The ideas of resetting and of the Newton-Raphson iteration can also be applicable for nonlinear networks, which are approximated by piecewise linear segments. For each segment, the network is linear and thus the proposed method can be applied. The most critical step is to determine the time at which the calculated $\hat{v}(t)$ crosses from one linear segment to another. The appropriate break points t_{break} can be found by the Newton-Raphson iteration since the voltage (or current) values of such break points are known from the piecewise linear characteristics. The solution of the piecewise linear networks in the time domain is then essentially equivalent to the search for such t_k which correspond to successive break points [63].

3. The proposed method can provide sensitivities of the time domain response by using frequency domain sensitivity methods. Consider a system with an input $U(s)$ and transfer function $F(s)$. Then the output $V(s)$ is

$$V(s) = F(s)U(s) \tag{3.75}$$

The frequency domain sensitivity with respect to any network parameter x, $\partial V/\partial x$, is obtained efficiently using the adjoint network approach [64]. Only the transpose of the admittance matrix is required to give the adjoint network equations [65]. Director and Rohrer also proposed a method for the calculation of time domain sensitivities [66] which, however, required integration and convolution [67].

When the Laplace transform inversion is used, convolution of the time domain functions becomes the product of the corresponding frequency domain functions. For the system (3.75) the time response is given by

$$\hat{v}(t) = -\frac{1}{t}\sum_{i=1}^{M} K_i F\left(\frac{z_i}{t}\right) U\left(\frac{z_i}{t}\right) \tag{3.76}$$

Let $F(s)$ depend on some parameter x, then

$$\frac{\partial \hat{v}(t)}{\partial x} = -\frac{1}{t}\sum_{i=1}^{M} K_i \frac{\partial F\left(\frac{z_i}{t}\right)}{\partial x} U\left(\frac{z_i}{t}\right) \tag{3.77}$$

The sensitivity is calculated from the sum of *M* frequency domain sensitivities where the complex values z_i/t are used instead of the actual frequencies. This is much simpler than convolution and integration.

3.8 Conclusion

This chapter, in summary, deals with the development of an analytic delay model for on-chip *RLC* interconnects. The proposed method uses numerical inversion of the Laplace transform to find the time domain response for linear distributed networks, which have infinite number of poles. The proposed method uses a Padé rational approximation to the exponential function and does not require the knowledge of the poles of the function under consideration. The degree of the approximation used determines the accuracy, and the poles and residues of the exponential function approximation can be computed with high

precision. In order to obtain the solution at a given time the frequency domain function has to be solved at a few complex points. The complex frequency points do not depend on the particular function under consideration and are pre-tabulated. The proposed method can be easily programmed. In each new problem, only the evaluation of the function has to be replaced.

The accuracy analysis of the proposed method is made from both the theoretical and empirical points of view. It is found that the accuracy is high for small time and that the proposed method is essentially equivalent to the numerical integration of differential equations that is stable for higher order approximation. In next chapter, the validity and efficiency of the proposed method will be verified by presenting numerical examples.

Chapter 4

Numerical Examples

This section presents three numerical examples to illustrate the validity and efficiency of the proposed work. Examples of single line, symmetrical unbalanced tree structure, and unsymmetrical tree structure are provided in this chapter to verify the advantages of the proposed work in terms of both accuracy and run time, respectively, when comparing with two-pole model [25] and Passive Reduced-Order Interconnect Macromodeling Algorithm, PRIMA [29]. In Section 4.1, the proposed method is verified when considering unit step signal input for the provided examples with interconnect structures varying from simple single line to complicated tress structures. In Section 4.2, the examples are verified when unit ramp signal input with rising time of 0.1 ns and 0.025 ns is selected to observe the results. The concluding remarks are presented in Section 4.3. The results are implemented using MATLAB R2018a [68] and are compared with HSPICE [69] simulation, two-pole model [25] and PRIMA [29]. The two-pole model of [25] is selected since approximating the cosh and sinh terms of (2.13) with a Maclaurin series results in more accurate models when compared to lumped *RLC* two-pole approximations. The PRIMA, which is an advanced method and is known to provide accurate *RLC* reduced models with different approximation orders without numerical issues, is selected as a comparison to the proposed method in respect to both accuracy and run time. To verify the accuracy of the NILT, the HSPICE simulations use 200 uniform lumped *RLC* segments to model each interconnect.

4.1 Selecting Unit Step Signal Input

In this section, the unit step signal input is considered for three numerical examples of single line, which correspond to a point-to-point interconnect, symmetrical unbalanced tree structure and unsymmetrical tree structure, respectively. The results obtained by the proposed method are compared to those of two-pole model [25], PRIMA [29], and HSPICE.

4.1.1 Example 1- Single Line Interconnect

4.1.1.1 Comparing the Proposed Algorithm to Two-Pole Model

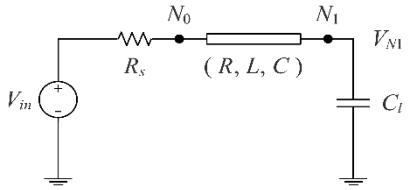


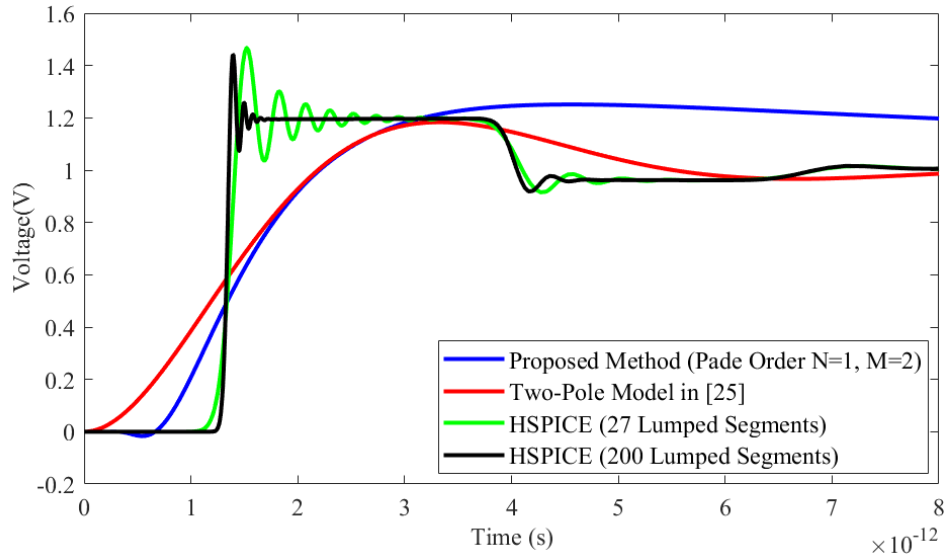
Figure 4. 1: Circuit model for a single transmission line

Table 4. 1: Per-unit-length parameters used for Example 1

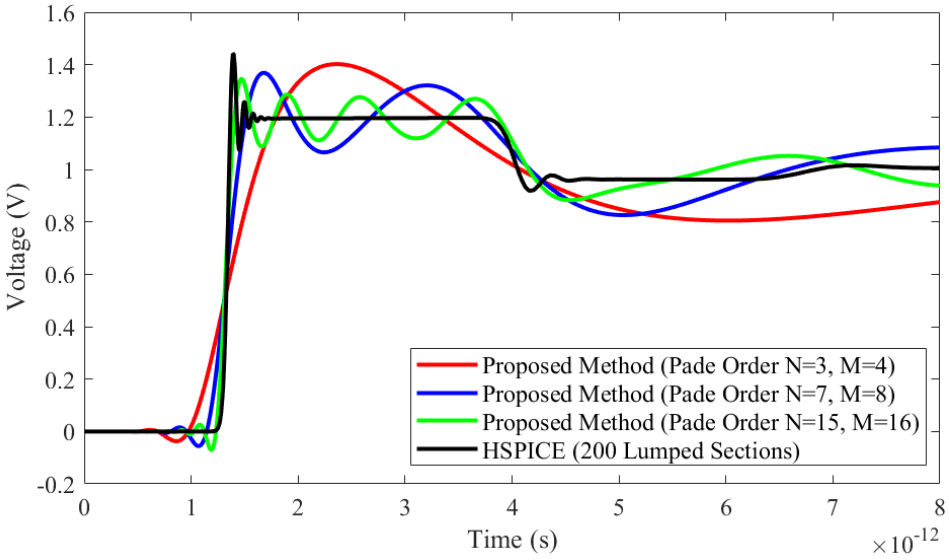
R (Ω/μm)	L (pH/μm)	C (fF/μm)
0.008829	1.538	0.18
0.0015	0.246	0.176

The interconnect network of Figure 4.1 is analyzed where the per-unit-length parameters are obtained from [1] and are listed in Table 4.1. The line length is set first to 0.02 cm and then is increased to 0.2 cm and the input voltage is the unit step response. The 10%, 50% and 90% delays at node N_1 calculated using the NILT are compared to HSPICE, two-pole model [25] and PRIMA [29] for various resistive and capacitive loads. The results of 10%, 50% and 90% delays at node N_1 are shown in Table 4.2 when comparing the proposed method with two-pole model [25]. Figure 4.2 (a) shows the transient responses comparing NILT, Padé order (N = 1, M = 2), HSPICE and the two-pole model for R = 0.0015 Ω/μm, L = 0.246 pH/μm, C = 0.176 fF/μm, R_s = 25 Ω, and C_l = 0.01 fF. Figure 4.2 (b) shows the NILT responses of the higher order Padé approximations. It is observed from Figure 4.2 that the NILT algorithm is accurate at time equal to zero and less accurate as time increases, while the two-pole model is accurate at steady state and less accurate in early time. For this example, Padé order (N = 1, M = 2) has average errors of 29.3%, 20.5%, and 45.9% for the 10%, 50% and 90% delay estimates, respectively, while the two-pole model has average errors of 60.9%, 24.4% and 51.0%, respectively (Table 4.2). Table 4.2 also provides 10%, 50% and 90% delay estimates for Padé order (N = 3, M = 4), (N = 7, M = 8) and (N = 15, M = 16), illustrating that as the order of the Padé approximation increases, the accuracy of the delay estimates improve. As seen from Table 4.2, Padé order (N = 15, M = 16) has the lowest average errors of 1.8%, 1.1%, and 1.6% for the 10%, 50% and 90% delay estimates,

respectively.



(a)



(b)

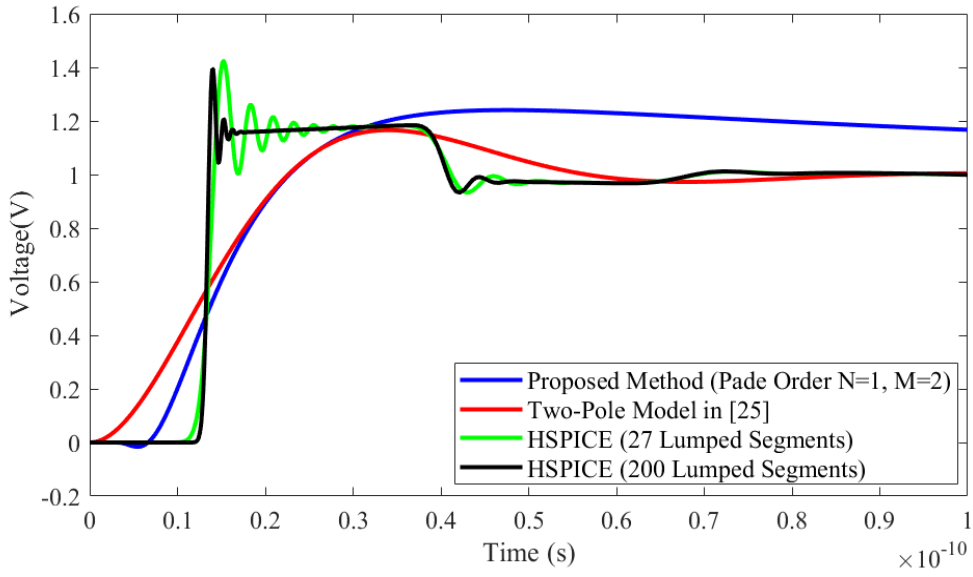
Figure 4. 2: The far end time domain response at node N_1 for Example 1. Line length = 0.02 cm, per-unit-length parameters (R = 0.0015 Ω/μm, L = 0.246 pH/μm, C = 0.176 fF/μm), R_s = 25 Ω, C_l = 0.01 fF (a) Comparison of proposed method Padé order (N = 1, M = 2), HSPICE and two-pole model [25] (b) Comparison of proposed method Padé orders (N = 3, M = 4), (N = 7, M = 8) and (N = 15, M = 16), respectively, against HSPICE

Table 4. 2: Comparisons of 10%, 50% and 90% delays at node N_1 using the proposed algorithm and the two-pole model [25] for Example 1, when line length is 0.02 cm

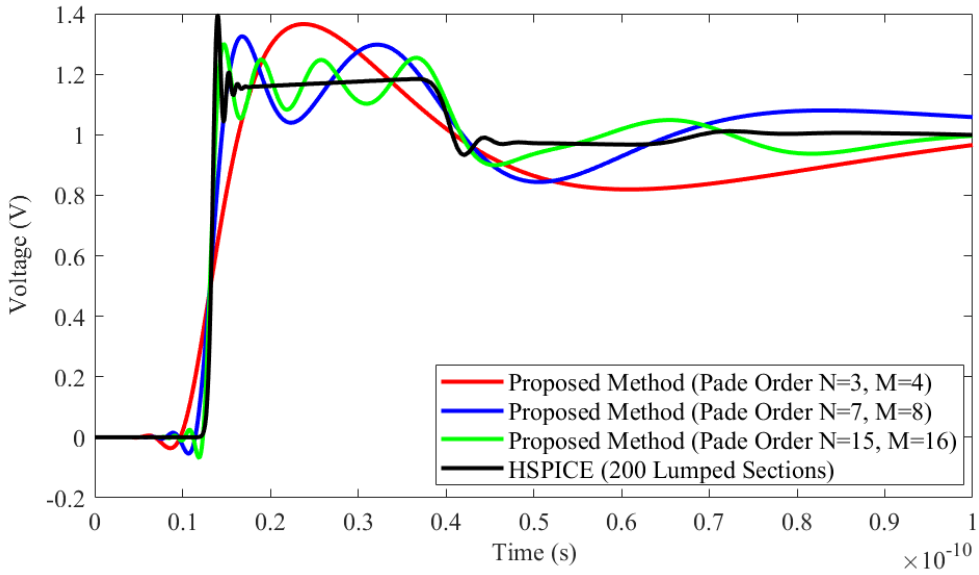
Per-unit-length parameters R(Ω/μm) L(pH/μm) C(fF/μm)	R_s (Ω)	C_l (fF)	HSPICE 10% delay (ps)	HSPICE 50% delay (ps)	HSPICE 90% delay (ps)	Two-Pole Model [25] 10% delay (ps)	Two-Pole Model [25] 50% delay (ps)	Two-Pole Model [25] 90% delay (ps)	Proposed Method (Padé Order N, M) (1, 2) 10% delay (ps)	(1, 2) 50% delay (ps)	(1, 2) 90% delay (ps)	(3, 4) 10% delay (ps)	(3, 4) 50% delay (ps)	(3, 4) 90% delay (ps)	(7, 8) 10% delay (ps)	(7, 8) 50% delay (ps)	(7, 8) 90% delay (ps)	(15, 16) 10% delay (ps)	(15, 16) 50% delay (ps)	(15, 16) 90% delay (ps)
R=0.008829 L=1.538 C=0.18	25	0.01	3.12	3.30	3.40	1.14	2.76	4.08	2.09	3.04	4.01	2.67	3.16	3.57	2.99	3.23	3.42	3.16	3.28	3.37
	50	0.01	3.19	3.33	3.44	1.14	2.91	4.56	2.15	3.28	4.63	2.70	3.27	3.79	3.02	3.28	3.52	3.17	3.30	3.41
	100	0.01	3.25	3.37	3.44	1.22	3.51	6.78	2.26	3.82	6.46	2.76	3.49	4.35	3.04	3.38	3.75	3.18	3.35	3.52
	25	0.1	3.13	3.31	3.40	1.14	2.70	3.96	2.09	3.04	4.02	2.67	3.16	3.58	3.00	3.24	3.43	3.17	3.28	3.38
	50	0.1	3.14	3.34	3.45	1.14	2.91	4.56	2.15	3.29	4.64	2.71	3.27	3.80	3.02	3.29	3.53	3.18	3.31	3.42
	100	0.1	3.26	3.38	3.45	1.22	3.51	6.80	2.27	3.83	6.48	2.77	3.50	4.36	3.05	3.39	3.76	3.19	3.36	3.53
R=0.0015 L=0.246 C=0.176	25	0.01	1.21	1.33	1.35	0.50	1.21	1.98	0.86	1.34	1.96	1.07	1.31	1.54	1.19	1.31	1.41	1.25	1.31	1.36
	50	0.01	1.25	1.33	1.35	0.50	1.52	2.09	0.92	1.62	2.04	1.10	1.42	1.87	1.21	1.36	1.54	1.26	1.33	1.42
	100	0.01	1.30	1.37	6.89	0.60	2.55	7.79	1.01	2.40	6.62	1.16	1.69	8.34	1.23	1.47	7.40	1.27	1.39	6.96
	25	0.1	1.29	1.33	1.35	0.46	1.20	1.94	0.86	1.35	1.97	1.08	1.31	1.55	1.20	1.31	1.41	1.25	1.31	1.36
	50	0.1	1.29	1.34	1.38	0.50	1.53	3.39	0.92	1.62	3.04	1.11	1.42	1.87	1.21	1.36	1.55	1.26	1.34	1.42
	100	0.1	1.29	1.34	6.91	0.60	2.56	7.82	1.02	2.41	6.64	1.16	1.70	8.36	1.24	1.48	7.42	1.28	1.39	6.97
Average Error (%)						60.9	24.4	51.0	29.3	20.5	45.9	13.6	7.2	19.0	4.8	2.7	6.2	1.8	1.1	1.6
Maximum Error (%)						64.3	91.0	145.7	33.3	79.9	120.3	16.3	26.9	38.5	7.0	10.4	14.1	3.3	3.7	5.2

To examine the robustness of the proposed algorithm for electrically longer lines, the line length of this example is increased to 0.2 cm. The 10%, 50% and 90% delays at node N_1 are calculated for the unit step response and the results are shown in Table 4.3. Figure 4.3 (a) shows the transient response comparing NILT, Padé order (N = 1, M = 2), HSPICE and the two-pole model for R = 0.0015 Ω/μm, L = 0.246 pH/μm, C = 0.176 fF/μm, R_s = 25 Ω, and C_l = 0.01 fF, while Figure 4.3 (b) shows the NILT responses of higher order Padé approximations. Once again, the NILT algorithm is able to improve the accuracy of the delay estimates by increasing the order of the Padé approximation. From Table 4.3, the accuracies of 10%, 50% and 90% delay estimations generated from both the proposed method and two-pole model become degraded due to the fact that longer transmission line length leads to larger signal delay. For the two-pole model, the average error for 10% delay increases from 60.9% to 61.1%, the average error for 50% delay increases from 24.4% to 25.1%, and the average error for 90% delay increases from 51.0% to 65.1%, respectively. For the proposed method, the average error for 10% delay increases from 29.3% to 30.4%, the average error for 50% delay increases from 20.5% to 22.3%, and the average error for 90% delay increases from 45.9% to 57.9%, respectively, when the Padé order is selected (N = 1, M = 2) (Table 4.3). By comparison, the proposed method is still capable of providing more accurate estimations of 10%, 50% and 90% than the two-pole model (Table 4.3) even for the electrically longer lines. In this case, the Padé order (N = 15, M = 16) results in the lowest average errors of 2.0%, 1.4% and 1.8% for the 10%, 50% and 90% delay estimates with the maximum errors of 2.5%, 7.8% and 3.8%, respectively.

To verify the efficiency of the proposed method, the run times to solve (3.36) for node N_1 at 500 time points are calculated and results are compared with those of the two-pole model. The average run times are 4.08 ms for the proposed algorithm with Padé order (N = 1, M = 2) and 0.69 ms for the two-pole model, respectively, when the transmission line length is 0.02 cm. When the line length is increased to 0.2 cm, the average run times are 3.72 ms for the proposed algorithm with Padé order (N = 1, M = 2) and 0.72 ms for the two-pole model, respectively. The comparative result shows that the two-pole model requires less run time than the proposed method, and the two-pole model is about 5 and 6 times faster than the proposed method for line length 0.02 cm and 0.2 cm, respectively.



(a)



(b)

Figure 4. 3: The far end time domain response at node N_1 for Example 1. Line length = 0.2 cm, per-unit-length parameters (R = 0.0015 Ω/µm, L = 0.246 pH/µm, C = 0.176 fF/µm), R_s = 25 Ω, C_l = 0.01 fF (a) Comparison of proposed method Padé order (N = 1, M = 2), HSPICE and two-pole model [25] (b) Comparison of proposed method Padé orders (N = 3, M = 4), (N = 7, M = 8) and (N = 15, M = 16), respectively, against HSPICE

Table 4. 3: Comparisons of 10%, 50% and 90% delays at node N_1 using the proposed algorithm and the two-pole model [25] for Example 1, when line length is 0.2 cm

Per-unit-length parameters R(Ω/μm) L(pH/μm) C(fF/μm)	R_s (Ω)	C_l (fF)	HSPICE 10% delay (ps)	HSPICE 50% delay (ps)	HSPICE 90% delay (ps)	Two-pole model [25] 10% delay (ps)	Two-pole model [25] 50% delay (ps)	Two-pole model [25] 90% delay (ps)	Proposed Method (Padé Order N, M) (1, 2) 10% delay (ps)	(1, 2) 50% delay (ps)	(1, 2) 90% delay (ps)	(3, 4) 10% delay (ps)	(3, 4) 50% delay (ps)	(3, 4) 90% delay (ps)	(7, 8) 10% delay (ps)	(7, 8) 50% delay (ps)	(7, 8) 90% delay (ps)	(15, 16) 10% delay (ps)	(15, 16) 50% delay (ps)	(15, 16) 90% delay (ps)
R=0.008829 L=1.538 C=0.18	25	0.01	32.36	33.30	33.77	11.50	28.00	41.50	21.13	31.31	42.19	26.82	32.00	36.59	29.99	32.52	34.62	31.63	32.86	33.85
	50	0.01	32.41	33.44	34.01	12.00	31.20	49.50	21.79	33.93	49.12	27.17	33.19	39.04	30.16	33.08	35.68	31.71	33.13	34.36
	100	0.01	32.50	33.73	34.56	12.60	36.75	73.05	23.01	39.70	69.32	27.83	35.59	45.63	30.49	34.18	38.48	31.87	33.65	35.68
	25	0.1	32.38	33.25	34.55	11.15	28.00	41.50	21.14	31.32	42.20	26.82	32.01	36.60	30.00	32.53	34.62	31.64	32.86	33.86
	50	0.1	32.42	33.45	34.02	11.70	30.30	48.60	20.80	33.94	49.13	27.18	33.20	39.05	30.17	33.09	35.69	31.72	33.13	34.36
	100	0.1	32.51	33.74	34.56	12.60	36.75	73.05	21.02	39.71	69.34	27.84	35.60	45.64	30.50	34.19	38.49	31.88	33.66	35.69
R=0.0015 L=0.246 C=0.176	25	0.01	12.76	13.24	13.50	4.60	12.20	20.00	8.65	13.62	20.13	10.76	13.20	15.61	11.94	13.11	14.18	12.55	13.12	13.62
	50	0.01	12.83	13.40	13.82	5.00	15.60	34.90	9.21	16.47	31.26	11.07	14.33	19.26	12.09	13.63	15.70	12.62	13.36	14.35
	100	0.01	12.93	13.77	69.66	6.20	25.80	78.80	10.23	24.43	67.16	11.60	17.21	84.33	12.35	14.85	74.37	12.75	13.93	70.37
	25	0.1	12.77	13.24	13.51	4.60	12.20	20.00	8.66	13.62	20.14	10.77	13.20	15.62	11.94	13.12	14.18	12.45	13.12	13.62
	50	0.1	12.83	13.40	13.82	5.10	15.60	34.90	9.22	16.47	31.27	11.07	14.34	19.26	12.09	13.63	15.70	12.59	13.36	14.35
	100	0.1	12.93	12.93	69.67	6.20	25.80	78.80	10.24	24.44	67.18	11.60	17.21	84.35	12.35	14.86	74.39	12.70	13.94	70.39
Average Error (%)						61.1	25.1	65.1	30.4	22.3	57.9	14.6	7.7	21.7	6.2	3.1	7.2	2.0	1.4	1.8
Maximum Error (%)						65.6	99.5	152.5	35.8	89.0	126.3	17.2	33.1	39.4	7.4	14.9	13.6	2.5	7.8	3.8

Though the two-pole model has lower run time compared to the proposed method, the two-pole model cannot be guaranteed always stable mathematically. Theoretically, the two-pole model, which is based on the second-order approximation of moment matching technique, will be instable when the values of b_2 in formula (2.69) is minus due to the large value of the inductance, shown as in Figure 4.4. In Figure 4.4, the first coefficient b_1 of formula (2.69) is independent of the inductance of transmission line and the second coefficient b_2 increases with the increases of transmission line inductance [25]. As the comparison, the proposed method is based on the numerical inversion of the Laplace transform (NILT) which was previously used as an integration formula for SPICE analysis of interconnect circuits. The NILT directly uses the frequency solution of interconnect circuits. Since NILT is equivalent to an integration formula that is stable for higher order approximations, the proposed algorithm provides a mechanism to increase the accuracy of the model for electrically long *RLC* interconnect circuits, where there are no stability and numerical problems such as suffered by the two-pole model.

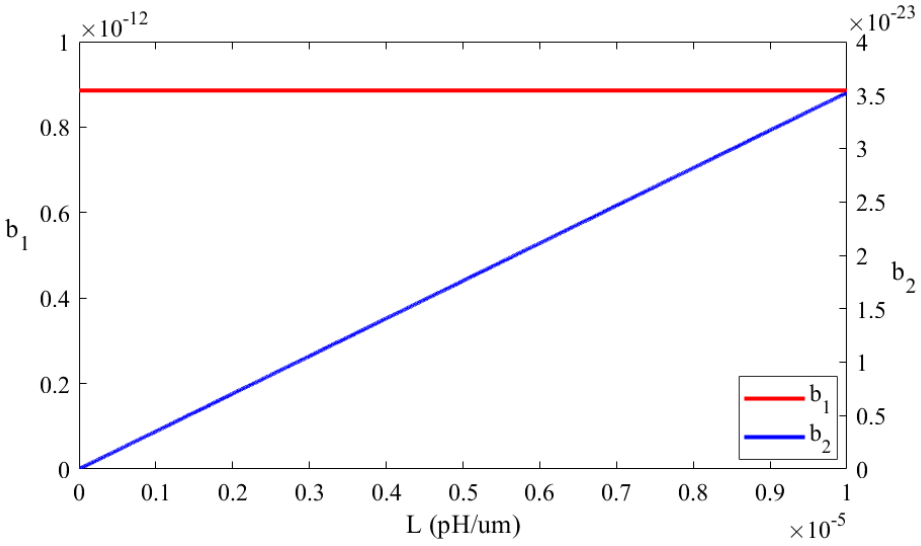


Figure 4. 4: Relationship between coefficients b_1, b_2 and inductance L of transmission line

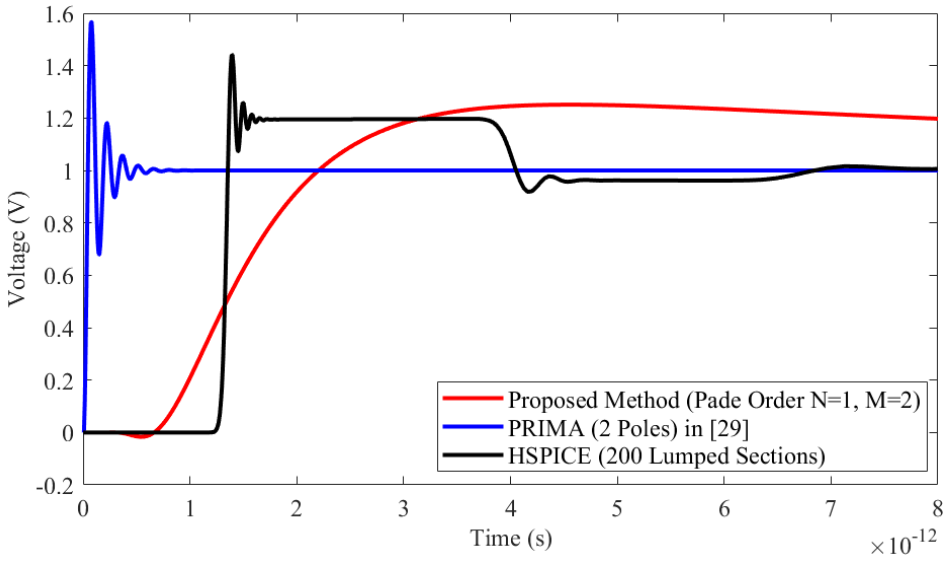
4.1.1.2 Comparing the Proposed Algorithm to PRIMA

Considering the limit of fixed approximation order of the two-pole model, accuracy and

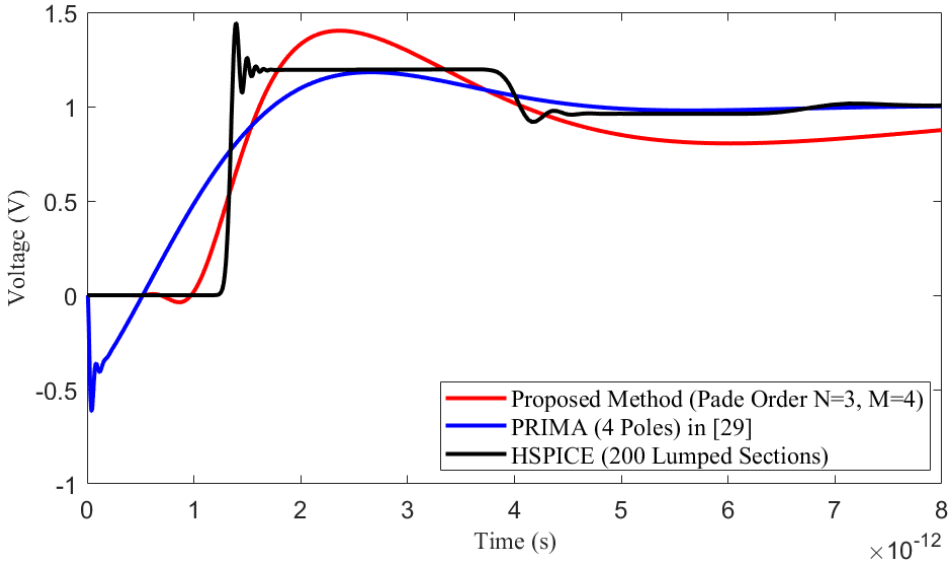
efficiency of the proposed method is verified further by comparing to the advanced moment matching based PRIMA, which matches moments by selecting different approximation orders. To keep the consistence in the Padé orders ((N = 1, M = 2), (N = 3, M = 4), (N = 7, M = 8) and (N = 15, M = 16)) of the proposed method, the approximation orders of the PRIMA are selected 2, 4, 8 and 16 for single line interconnect, which matches 1, 2, 4, 8 moments, respectively. The calculations of 10%, 50% and 90% delays at node N_1 using the NILT, HSPICE and PRIMA [29] for various resistive and capacitive loads are listed in Table 4.4 for line length of 0.02 cm. Figure 4.5 (a)-(b) show the transient responses obtained from the NILT with Padé orders (N = 1, M = 2) and (N = 3, M = 4), HSPICE and the PRIMA with approximation orders 2 and 4, respectively, for R = 0.0015 Ω/μm, L = 0.246 pH/μm, C = 0.176 fF/μm, R_s = 25 Ω, and C_l = 0.01 fF. Figure 4.5 (c)-(d) show the comparison of responses of the higher order approximations between the NILT (Padé orders (N = 7, M = 8) and (N = 15, M = 16), respectively) and the PRIMA (approximation orders 8 and 16, respectively) for the same per-unit-length parameters.

It is observed from Figure 4.5 that the NILT algorithm is accurate at time equal to zero and less accurate as time increases, while PRIMA is more accurate at steady state and less accurate in early time. For this example, the NILT with Padé order (N = 1, M = 2) has average errors of 29.3%, 20.5%, and 45.9% for the 10%, 50% and 90% delay estimates, respectively, while PRIMA has average errors of 97.7%, 95.5% and 94.8%, respectively when approximation order of 2 is considered (Table 4.4). Table 4.4 also provides 10%, 50% and 90% delay estimates for both the NILT with higher Padé orders of (N = 3, M = 4), (N = 7, M = 8) and (N = 15, M = 16), and PRIMA with higher approximation orders of 4, 8 and 16. As illustrated in Table 4.4 that as the Padé approximation order of the NILT increases, the accuracy of the delay estimates is improved greatly. As seen from Table 4.4, Padé order (N = 15, M = 16) for the NILT has the lowest average errors of 1.8%, 1.1%, and 1.6% for the 10%, 50% and 90% delay estimates. As a comparison, the accuracy of 10%, 50% and 90% delay estimates of PRIMA is improved significantly when the approximation order increases from 2 to 4 with the average error dropped to 54.6% from 97.7% for 10% delay, average error dropped to 31.3% from 95.5% for 50% delay, and average error dropped to 47.2% from 94.8% for 90% delay, respectively. The accuracy of 10% delay

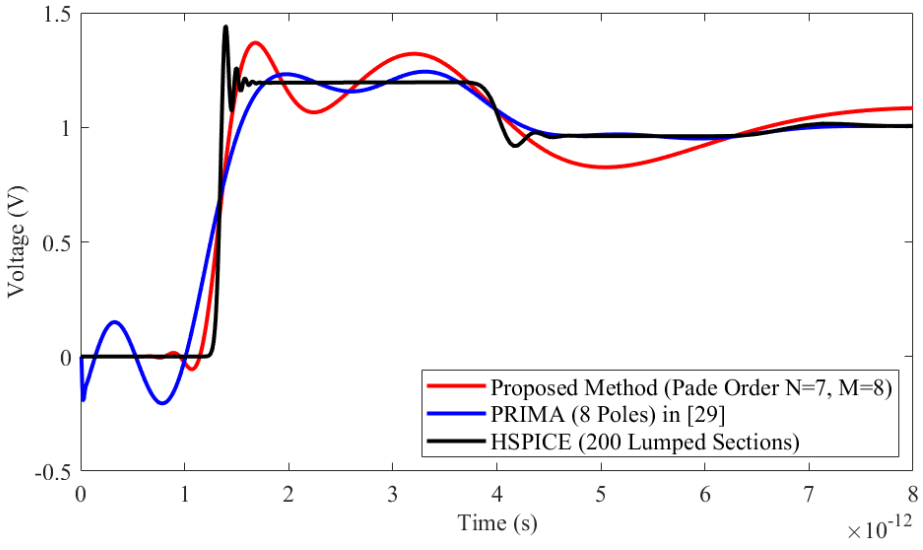
becomes a slightly worse when the approximation order increases to 8 while the accuracies of 50% and 90% delay estimations are improved compared to lower approximation orders. The accuracy of steady state is improved greatly when the highest approximation order of 16 is selected for PRIMA, however, the accuracies of 10%, 50% and 90% delay estimations are kept unimproved compared with lower approximation order of 8, both of which have the same average errors of 59.2%, 11.2% and 40.1% for 10%, 50% and 90% delay estimations, respectively.



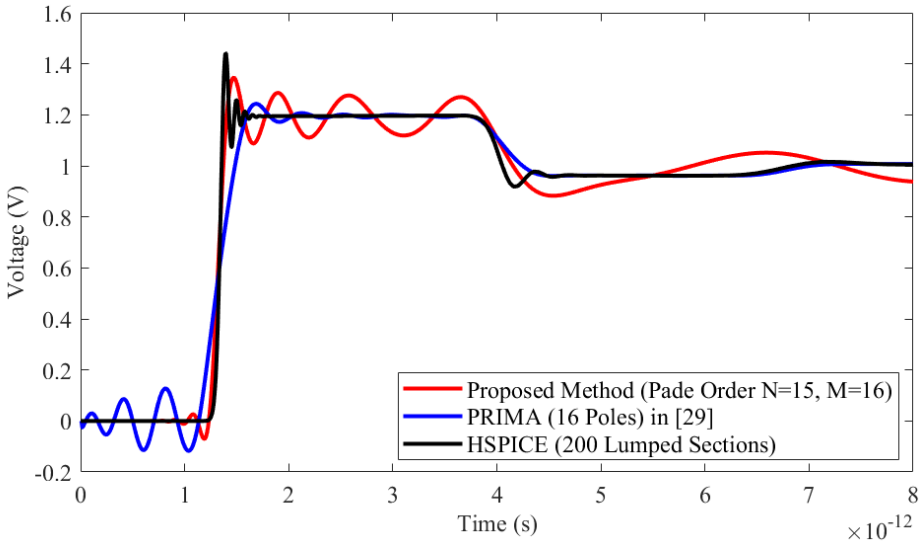
(a)



(b)



(c)



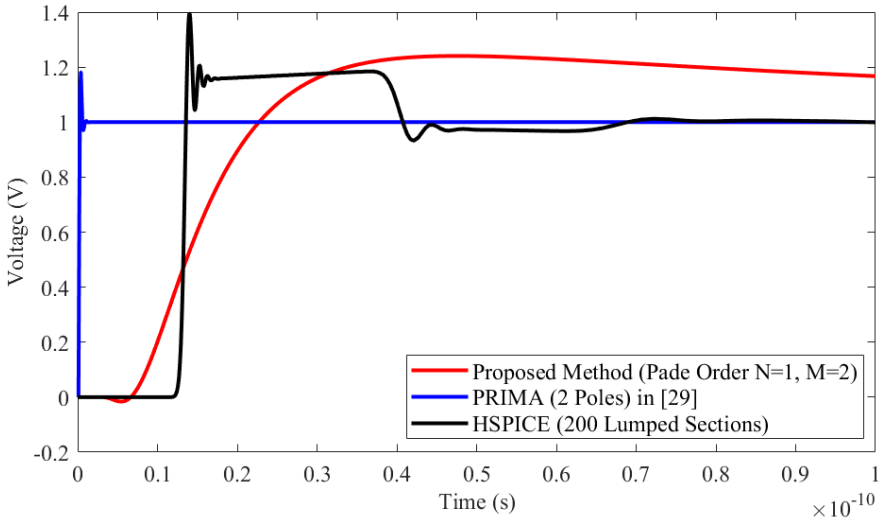
(d)

Figure 4. 5: The far end time domain response at node N_1 for Example 1. Line length = 0.02 cm, per-unit-length parameters (R = 0.0015 Ω/μm, L = 0.246 pH/μm, C = 0.176 fF/μm), R_s = 25 Ω, C_l = 0.01 fF (a) Comparison of proposed method Padé order (N = 1, M = 2), HSPICE and PRIMA (2 poles) [29] (b) Comparison of proposed method Padé order (N = 3, M = 4), HSPICE and PRIMA (4 poles) [29] (c) Comparison of proposed method Padé order (N = 7, M = 8), HSPICE and PRIMA (8 poles) [29] (d) Comparison of proposed method Padé order (N = 15, M = 16), HSPICE and PRIMA (16 poles) [29]

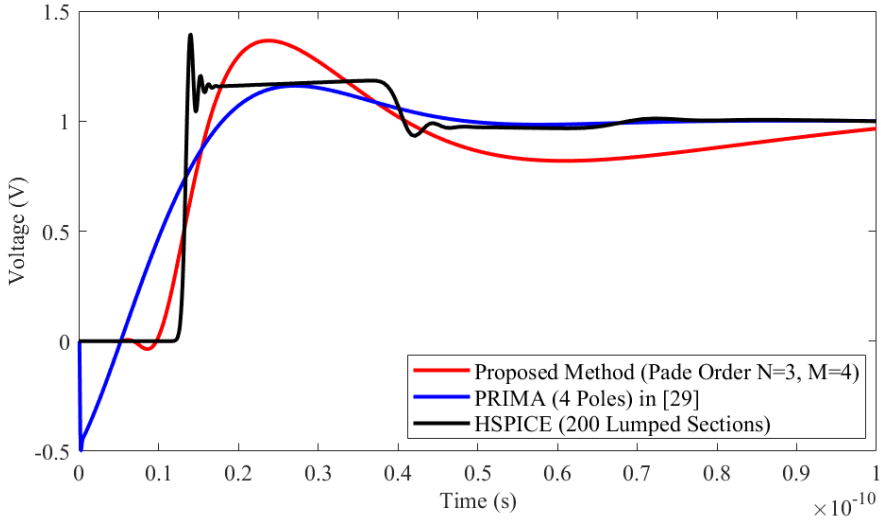
In the accuracy comparison of this example, the proposed algorithm can improve the accuracies of 10%, 50% and 90% delay estimations obviously as the Padé order increases, while PRIMA provide less accurate estimations for 10%, 50% and 90% delay and the accuracies of the 10%, 50% and 90% delay estimations are not improved though the approximation order is increased further which brings about more accurate steady state.

Increasing the line length to 0.2 cm to examine the robustness of the proposed algorithm for electrically longer lines when comparing with PRIMA [29]. The 10%, 50% and 90% delays at node N_1 are calculated for the unit step response and the results are shown in Table 4.5. Figure 4.6 (a) shows the transient response comparing NILT, Padé order (N = 1, M = 2), HSPICE and PRIMA with the approximation order of 2 for R = 0.0015 Ω/μm, L = 0.246 pH/μm, C = 0.176 fF/μm, R_s = 25 Ω, and C_l = 0.01 fF, while Figure 4.6 (b)-(d) show the NILT responses of higher Padé orders and PRIMA responses of higher approximation orders, respectively. Due to the larger signal delay caused by increasing line length from 0.02 cm to 0.2 cm, the accuracies of 10%, 50% and 90% delay estimations obtained from both the proposed algorithm and PRIMA become worse. As observed in Table 4.5, increasing the approximation orders of PRIMA does not contribute to improve the accuracies of 10%, 50% and 90% delay estimations significantly for unit step input and electrically longer line though increasing approximation orders improves the accuracy of steady state. For approximation orders of 8 and 16, the accuracies of 10%, 50% and 90% delay estimations keep unimproved. Compared to PRIMA, the NILT based proposed algorithm is able to improve the accuracy of the delay estimates greatly once again by increasing the order of the Padé approximation. For PRIMA with approximation order of 2, the average error for 10% delay increases from 97.7% to 99.8%, the average error for 50% delay increases from 95.5% to 99.5%, and the average error for 90% delay increases from 94.8% to 99.5%, respectively when the line length increases from 0.02 cm to 0.2 cm. For the proposed method with Padé order (N = 1, M = 2), the average error for 10% delay increases from 29.3% to 30.4%, the average error for 50% delay increases from 20.5% to 22.3%, and the average error for 90% delay increases from 45.9% to 57.9%, respectively, when the line length increases from 0.02 cm to 0.2 cm (Table 4.5). When selecting the highest approximation order of 16 for PRIMA, the average error for 10%

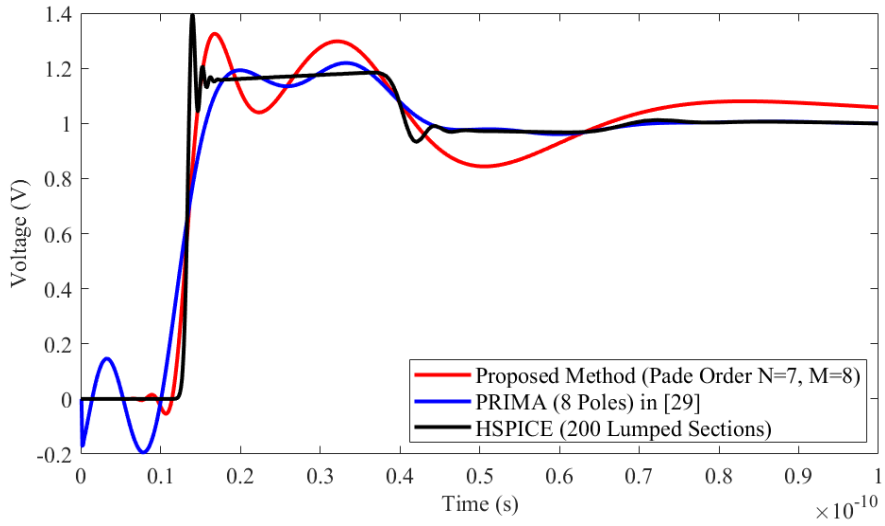
delay only drops from 99.8% to 96.0%, the average error for 50% delay drops from 99.5% to 90.4%, and the average error for 90% delay drops from 99.5% to 86.9%, respectively compared with the lowest approximation order of 2. For the proposed algorithm, the average error for 10% delay drops from 30.4% to 2.0%, the average error for 50% delay drops from 22.3% to 1.4%, and the average error for 90% delay drops from 57.9% to 1.8%, respectively when increasing the Padé order from (N = 1, M = 2) to (N = 15, M = 16). By comparison, the proposed algorithm is still able to generate more accurate estimations of 10%, 50% and 90% than PRIMA and the accuracies of delay estimates can be improved significantly as the Padé order increases (Table 4.5) for the electrically longer lines.



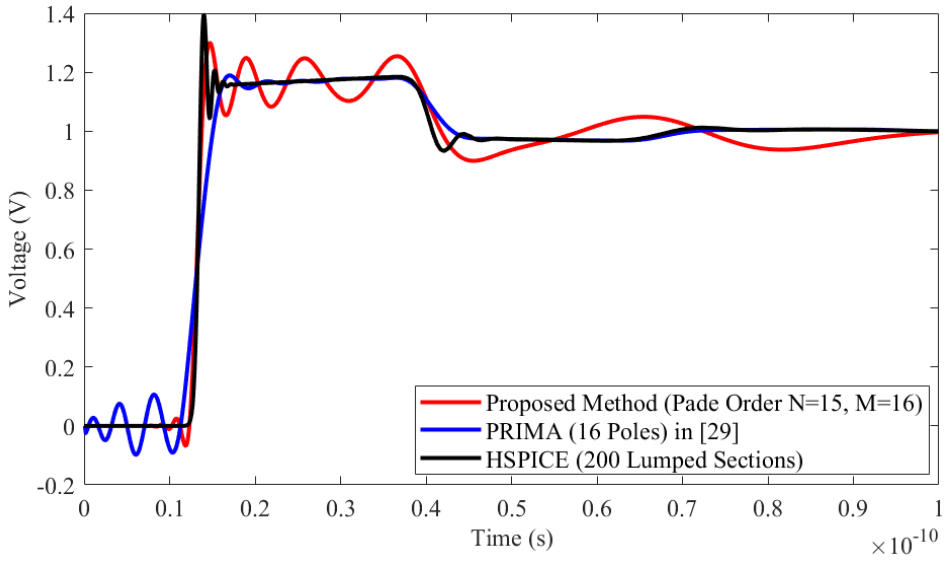
(a)



(b)



(c)



(d)

Figure 4. 6: The far end time domain response at node N_1 for Example 1. Line length = 0.2 cm, per-unit-length parameters (R = 0.0015 Ω/µm, L = 0.246 pH/µm, C = 0.176 fF/µm), R_s = 25 Ω, C_l = 0.01 fF (a) Comparison of proposed method Padé order (N = 1, M = 2), HSPICE and PRIMA (2 poles) [29] (b) Comparison of proposed method Padé order (N = 3, M = 4), HSPICE and PRIMA (4 poles) [29] (c) Comparison of proposed method Padé order (N = 7, M = 8), HSPICE and PRIMA (8 poles) [29] (d) Comparison of proposed method Padé order (N = 15, M = 16), HSPICE and PRIMA (16 poles) [29]

Table 4. 4: Comparisons of 10%, 50% and 90% delays at node N_1 using the proposed method and PRIMA [29] for Example 1, when line length is 0.02 cm

Per-unit-length parameters R(Ω/μm) L(pH/μm) C(fF/μm)	R_s (Ω)	C_l (fF)	HSPICE			PRIMA [29] (Approximation Order)						Proposed Method (Padé Order N, M)					
						2			4			(1, 2)			(3, 4)		
			10% delay (ps)	50% delay (ps)	90% delay (ps)	10% delay (ps)	50% delay (ps)	90% delay (ps)	10% delay (ps)	50% delay (ps)	90% delay (ps)	10% delay (ps)	50% delay (ps)	90% delay (ps)	10% delay (ps)	50% delay (ps)	90% delay (ps)
R=0.008829 L=1.538 C=0.18	25	0.01	3.12	3.30	3.40	0.075	0.12	0.16	1.73	2.49	3.25	2.09	3.04	4.01	2.67	3.16	3.57
	50	0.01	3.19	3.33	3.44	0.047	0.086	0.12	0.86	1.24	1.63	2.15	3.28	4.63	2.70	3.27	3.79
	100	0.01	3.25	3.37	3.44	0.035	0.071	0.098	1.47	2.90	6.09	2.26	3.82	6.46	2.76	3.49	4.35
	25	0.1	3.13	3.31	3.40	0.12	0.23	0.33	1.73	2.49	3.26	2.09	3.04	4.02	2.67	3.16	3.58
	50	0.1	3.14	3.34	3.45	0.098	0.21	0.29	1.59	2.54	3.63	2.15	3.29	4.64	2.71	3.27	3.80
	100	0.1	3.26	3.38	3.45	0.089	0.20	0.28	1.09	2.98	6.10	2.27	3.83	6.48	2.77	3.50	4.36
R=0.0015 L=0.246 C=0.176	25	0.01	1.21	1.33	1.35	0.016	0.030	0.042	0.62	1.02	1.54	0.86	1.34	1.96	1.07	1.31	1.54
	50	0.01	1.25	1.33	1.35	0.017	0.032	0.045	0.57	1.29	3.35	0.92	1.62	2.04	1.10	1.42	1.87
	100	0.01	1.30	1.37	6.89	0.022	0.038	0.052	0.56	2.44	7.83	1.01	2.40	6.62	1.16	1.69	8.34
	25	0.1	1.29	1.33	1.35	0.036	0.080	0.11	0.61	1.03	1.54	0.86	1.35	1.97	1.08	1.31	1.55
	50	0.1	1.29	1.34	1.38	0.037	0.081	0.11	0.59	1.28	3.35	0.92	1.62	3.04	1.11	1.42	1.87
	100	0.1	1.29	1.34	6.91	0.045	0.093	0.13	0.58	2.45	7.86	1.02	2.41	6.64	1.16	1.70	8.36
Average Error (%)						97.7	95.5	94.8	54.6	31.3	47.2	29.3	20.5	45.9	13.6	7.2	19.0
Maximum Error (%)						99.9	97.9	99.2	73.0	82.8	148.1	33.3	79.9	120.3	16.3	26.9	38.5

Table 4.4 (Cont'd.):

Per-unit-length parameters R(Ω/μm) L(pH/μm) C(fF/μm)	R_s (Ω)	C_l (fF)	HSPICE 10% delay (ps)	HSPICE 50% delay (ps)	HSPICE 90% delay (ps)	PRIMA [29] (Approximation Order) 8 10% delay (ps)	PRIMA [29] (Approximation Order) 8 50% delay (ps)	PRIMA [29] (Approximation Order) 8 90% delay (ps)	PRIMA [29] (Approximation Order) 16 10% delay (ps)	PRIMA [29] (Approximation Order) 16 50% delay (ps)	PRIMA [29] (Approximation Order) 16 90% delay (ps)	Proposed Method (Padé Order N, M) (7, 8) 10% delay (ps)	Proposed Method (Padé Order N, M) (7, 8) 50% delay (ps)	Proposed Method (Padé Order N, M) (7, 8) 90% delay (ps)	Proposed Method (Padé Order N, M) (15, 16) 10% delay (ps)	Proposed Method (Padé Order N, M) (15, 16) 50% delay (ps)	Proposed Method (Padé Order N, M) (15, 16) 90% delay (ps)
R=0.008829 L=1.538 C=0.18	25	0.01	3.12	3.30	3.40	0.62	3.04	3.45	0.62	3.04	3.45	2.99	3.23	3.42	3.16	3.28	3.37
	50	0.01	3.19	3.33	3.44	0.31	1.52	1.73	0.31	1.52	1.73	3.02	3.28	3.52	3.17	3.30	3.41
	100	0.01	3.25	3.37	3.44	0.60	3.31	4.10	0.60	3.31	4.10	3.04	3.38	3.75	3.18	3.35	3.52
	25	0.1	3.13	3.31	3.40	0.63	3.05	3.46	0.63	3.05	3.46	3.00	3.24	3.43	3.17	3.28	3.38
	50	0.1	3.14	3.34	3.45	0.57	3.13	3.63	0.57	3.13	3.63	3.02	3.29	3.53	3.18	3.31	3.42
	100	0.1	3.26	3.38	3.45	0.65	3.32	4.11	0.65	3.32	4.11	3.05	3.39	3.76	3.19	3.36	3.53
R=0.0015 L=0.246 C=0.176	25	0.01	1.21	1.33	1.35	0.23	1.25	1.46	0.23	1.25	1.46	1.19	1.31	1.41	1.25	1.31	1.36
	50	0.01	1.25	1.33	1.35	1.09	1.35	3.79	1.09	1.35	3.79	1.21	1.36	1.54	1.26	1.33	1.42
	100	0.01	1.30	1.37	6.89	1.14	1.61	7.28	1.14	1.61	7.28	1.23	1.47	7.40	1.27	1.39	6.96
	25	0.1	1.29	1.33	1.35	0.22	1.25	1.46	0.22	1.25	1.46	1.20	1.31	1.41	1.25	1.31	1.36
	50	0.1	1.29	1.34	1.38	1.09	1.35	3.80	1.09	1.35	3.80	1.21	1.36	1.55	1.26	1.34	1.42
	100	0.1	1.29	1.34	6.91	1.14	1.64	7.36	1.14	1.64	7.36	1.24	1.48	7.42	1.28	1.39	6.97
Average Error (%)						59.2	11.2	40.1	59.2	11.2	40.1	4.8	2.7	6.2	1.8	1.1	1.6
Maximum Error (%)						90.3	54.4	180.7	90.3	54.4	180.7	7.0	10.4	14.1	3.3	3.7	5.2

Table 4. 5: Comparisons of 10%, 50% and 90% delays at node N_1 using the proposed method and PRIMA [29] for Example 1, when line length is 0.2 cm

Per-unit-length parameters R(Ω/μm) L(pH/μm) C(fF/μm)	R_s	C_l	HSPICE			PRIMA [29] (Approximation Order)						Proposed Method (Padé Order N, M)					
						2			4			(1, 2)			(3, 4)		
			10% delay	50% delay	90% delay	10% delay	50% delay	90% delay	10% delay	50% delay	90% delay	10% delay	50% delay	90% delay	10% delay	50% delay	90% delay
	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
R=0.008829 L=1.538 C=0.18	25	0.01	32.36	33.30	33.77	0.075	0.12	0.16	1.73	2.49	3.25	21.13	31.31	42.19	26.82	32.00	36.59
	50	0.01	32.41	33.44	34.01	0.047	0.086	0.12	0.86	1.24	1.63	21.79	33.93	49.12	27.17	33.19	39.04
	100	0.01	32.50	33.73	34.56	0.035	0.071	0.098	1.47	2.90	6.09	23.01	39.70	69.32	27.83	35.59	45.63
	25	0.1	32.38	33.25	34.55	0.12	0.23	0.33	1.73	2.49	3.26	21.14	31.32	42.20	26.82	32.01	36.60
	50	0.1	32.42	33.45	34.02	0.098	0.21	0.29	1.59	2.54	3.63	20.80	33.94	49.13	27.18	33.20	39.05
	100	0.1	32.51	33.74	34.56	0.089	0.20	0.28	1.09	2.98	6.10	21.02	39.71	69.34	27.84	35.60	45.64
R=0.0015 L=0.246 C=0.176	25	0.01	12.76	13.24	13.50	0.016	0.030	0.042	0.62	1.02	1.54	8.65	13.62	20.13	10.76	13.20	15.61
	50	0.01	12.83	13.40	13.82	0.017	0.032	0.045	0.57	1.29	3.35	9.21	16.47	31.26	11.07	14.33	19.26
	100	0.01	12.93	13.77	69.66	0.022	0.038	0.052	0.56	2.44	7.83	10.23	24.43	67.16	11.60	17.21	84.33
	25	0.1	12.77	13.24	13.51	0.036	0.080	0.11	0.61	1.03	1.54	8.66	13.62	20.14	10.77	13.20	15.62
	50	0.1	12.83	13.40	13.82	0.037	0.081	0.11	0.59	1.28	3.35	9.22	16.47	31.27	11.07	14.34	19.26
	100	0.1	12.93	12.93	69.67	0.045	0.093	0.13	0.58	2.45	7.86	10.24	24.44	67.18	11.60	17.21	84.35
Average Error (%)						99.8	99.5	99.5	95.5	90.4	86.4	30.4	22.3	57.9	14.6	7.7	21.7
Maximum Error (%)						99.9	99.8	99.9	97.3	96.3	95.2	35.8	89.0	126.3	17.2	33.1	39.4

Table 4.5 (Cont'd.):

Per-unit-length parameters R(Ω/μm) L(pH/μm) C(fF/μm)	R_s (Ω)	C_l (fF)	HSPICE 10% delay (ps)	HSPICE 50% delay (ps)	HSPICE 90% delay (ps)	PRIMA [29] (Approximation Order) 8 10% delay (ps)	PRIMA [29] (Approximation Order) 8 50% delay (ps)	PRIMA [29] (Approximation Order) 8 90% delay (ps)	PRIMA [29] (Approximation Order) 16 10% delay (ps)	PRIMA [29] (Approximation Order) 16 50% delay (ps)	PRIMA [29] (Approximation Order) 16 90% delay (ps)	Proposed Method (Padé Order N, M) (7, 8) 10% delay (ps)	Proposed Method (Padé Order N, M) (7, 8) 50% delay (ps)	Proposed Method (Padé Order N, M) (7, 8) 90% delay (ps)	Proposed Method (Padé Order N, M) (15, 16) 10% delay (ps)	Proposed Method (Padé Order N, M) (15, 16) 50% delay (ps)	Proposed Method (Padé Order N, M) (15, 16) 90% delay (ps)
R=0.008829 L=1.538 C=0.18	25	0.01	32.36	33.30	33.77	0.62	3.04	3.45	0.62	3.04	3.45	29.99	32.52	34.62	31.63	32.86	33.85
	50	0.01	32.41	33.44	34.01	0.31	1.52	1.73	0.31	1.52	1.73	30.16	33.08	35.68	31.71	33.13	34.36
	100	0.01	32.50	33.73	34.56	0.60	3.31	4.10	0.60	3.31	4.10	30.49	34.18	38.48	31.87	33.65	35.68
	25	0.1	32.38	33.25	34.55	0.63	3.05	3.46	0.63	3.05	3.46	30.00	32.53	34.62	31.64	32.86	33.86
	50	0.1	32.42	33.45	34.02	0.57	3.13	3.63	0.57	3.13	3.63	30.17	33.09	35.69	31.72	33.13	34.36
	100	0.1	32.51	33.74	34.56	0.65	3.32	4.11	0.65	3.32	4.11	30.50	34.19	38.49	31.88	33.66	35.69
R=0.0015 L=0.246 C=0.176	25	0.01	12.76	13.24	13.50	0.23	1.25	1.46	0.23	1.25	1.46	11.94	13.11	14.18	12.55	13.12	13.62
	50	0.01	12.83	13.40	13.82	1.09	1.35	3.79	1.09	1.35	3.79	12.09	13.63	15.70	12.62	13.36	14.35
	100	0.01	12.93	13.77	69.66	1.14	1.61	7.28	1.14	1.61	7.28	12.35	14.85	74.37	12.75	13.93	70.37
	25	0.1	12.77	13.24	13.51	0.22	1.25	1.46	0.22	1.25	1.46	11.94	13.12	14.18	12.45	13.12	13.62
	50	0.1	12.83	13.40	13.82	1.09	1.35	3.80	1.09	1.35	3.80	12.09	13.63	15.70	12.59	13.36	14.35
	100	0.1	12.93	12.93	69.67	1.14	1.64	7.36	1.14	1.64	7.36	12.35	14.86	74.39	12.70	13.94	70.39
Average Error (%)						96.0	90.4	86.9	96.0	90.4	86.9	6.2	3.1	7.2	2.0	1.4	1.8
Maximum Error (%)						99.0	95.5	94.9	99.0	95.5	94.9	7.4	14.9	13.6	2.5	7.8	3.8

To further compare the efficiency of the proposed method to that of PRIMA, 500 time points are selected to calculate the run times for solving the transient response at node N_1 obtained from both the proposed algorithm and PRIMA, respectively (Table 4.6). When using PRIMA for transient response analysis, a transmission line needs to be divided into numbers of lumped RLC sections, which determines the size of matrices used in modified nodal analysis (MNA) when characterizing the electrical behavior of interconnect transmission line. In order to obtain more accurate approximation, more moments are required to be matched, which requires more sections of lumped RLC. The size of matrices needed by PRIMA increases significantly due to the increasing number of lumped RLC sections. As a comparison, the NILT based proposed method does not need to divide the transmission line into numbers of lumped RLC sections, which directly uses the frequency solution of interconnect circuits based on Y-parameter (admittance), which has smaller size compared to the size of matrices in PRIMA. As shown in Table 4.6, the proposed method has linear complexity with the order of the Padé approximation and PRIMA requires much run time than the proposed method. For this example, the proposed algorithm is 16+ time faster than PRIMA for line length of 0.02 cm and is 23+ time faster than PRIMA for line length of 0.2 cm when comparing both methods at the same approximation order level.

Table 4. 6: Run time comparison between the proposed method and PRIMA [29] for Example 1

Length (cm)	PRIMA [29]		Proposed Method		Speed Up
	Approximation Order	Run Time (ms)	Padé Order	Run Time (ms)	
0.02	2	257.62	(1, 2)	4.08	63
	4	250.00	(3, 4)	7.40	34
	8	265.25	(7, 8)	10.99	24
	16	271.17	(15, 16)	16.94	16
0.2	2	260.86	(1, 2)	3.72	70
	4	258.97	(3, 4)	5.44	48
	8	288.06	(7, 8)	7.55	38
	16	303.58	(15, 16)	13.04	23

4.1.2 Example 2- Symmetrical Unbalanced Distributed RLC Tree

4.1.2.1 Comparing the Proposed Algorithm to Two-Pole Model

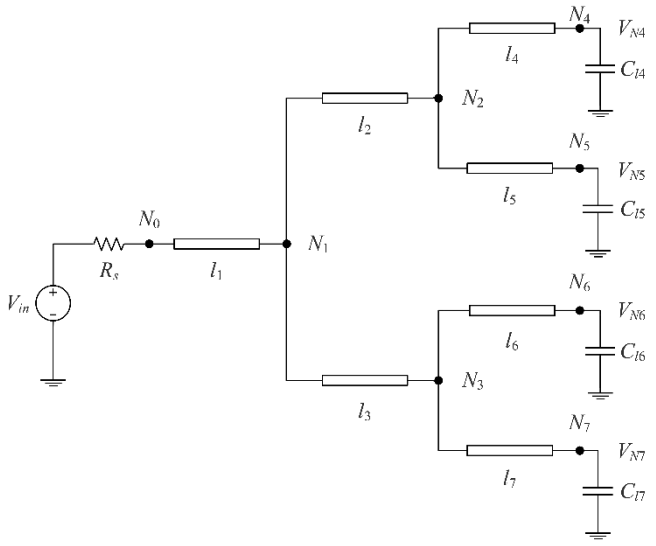


Figure 4. 7: General distributed *RCL* tree

Table 4. 7: Interconnect lengths normalized to l_x used for Example 2

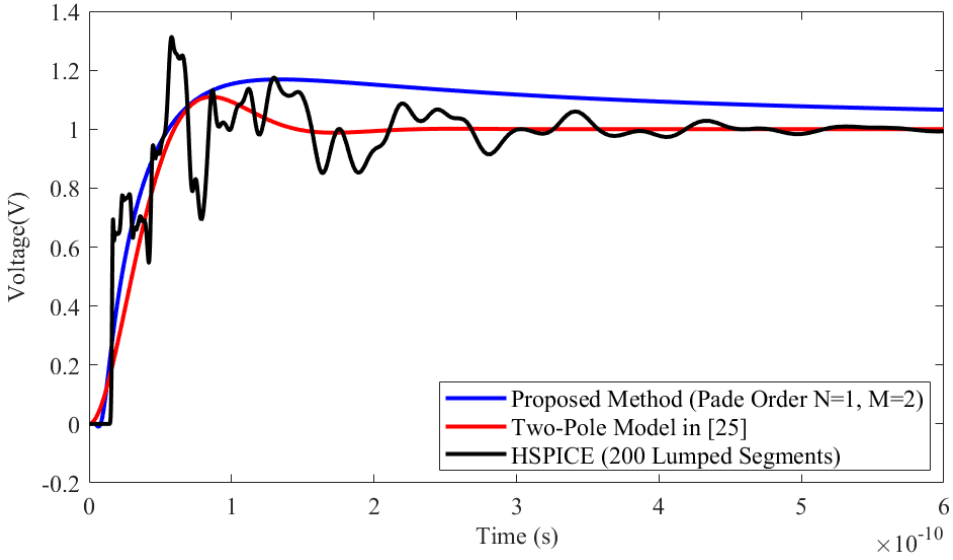
Index	l_1	l_2	l_3	l_4	l_5	l_6	l_7
Line length	0.25	0.5*x*	0.5	0.5*x*	0.5	0.5*x*	0.5

Table 4. 8: Load capacitances normalized to C_x used for Example 2

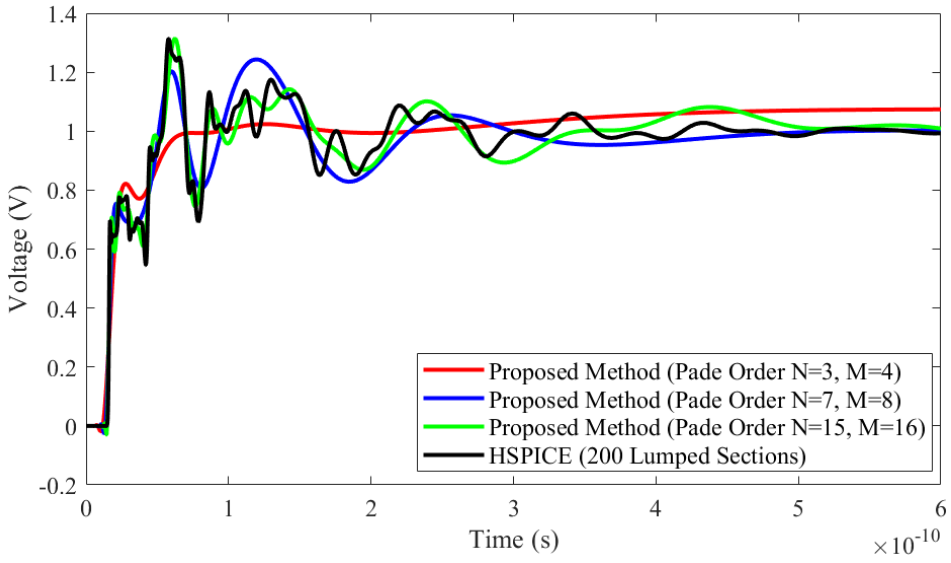
Index	C_{l4}	C_{l5}	C_{l6}	C_{l7}
Load capacitance	2	1	2	5

In this example, the proposed algorithm is demonstrated and compared with HSPICE, the two-pole model [25], and PRIMA [29], respectively, by considering the symmetrical unbalanced tree circuit of Figure 4.7. The branches in an interconnect tress structure can have various parasitic interconnect impedances. In this example, it is assumed that all the branches in the symmetrical unbalanced tree structure have the same width of 6 µm for the purpose of simplicity. The per-unit-length parameters are obtained from [33], where each line is characterized by R = 3.9 mΩ/µm, L = 0.43 pH/µm and C = 0.36 fF/µm. The normalized wire lengths and load capacitances are listed in Table 4.7 and Table 4.8, where l_x and C_x are the normalized reference length and capacitance, respectively. For the scenario

of symmetrical unbalanced tree interconnect structure, the variable l_x is set to one, x is set to two and four for different resistive load R_s and capacitive loads C_{lx}.



(a)



(b)

Figure 4. 8: The far end time domain response at node N_7 for Example 2 with unit step input, x = 2, R_s = 10 Ω, C_x= 20 fF (a) Comparison of proposed method Padé order (N = 1, M = 2), HSPICE and two-pole model [25] (b) Comparison of proposed method Padé orders (N = 3, M = 4), (N = 7, M = 8) and (N = 15, M = 16), respectively, against HSPICE

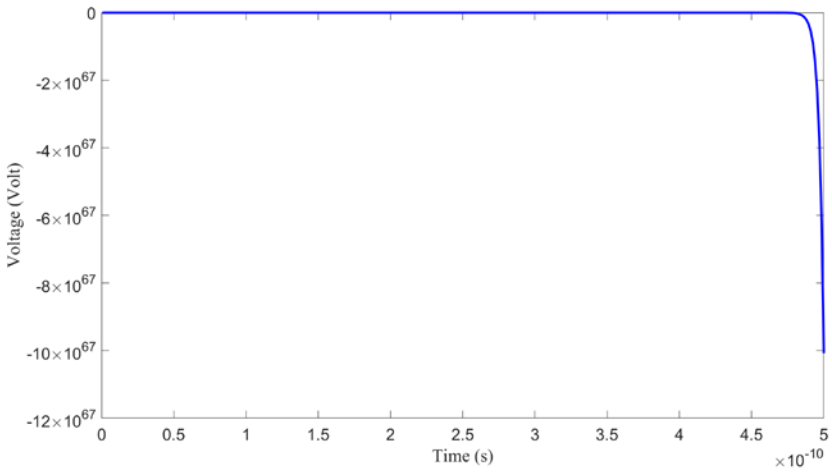


Figure 4. 9: The far end time domain response at N_7 with a unit step input using an unstable three-pole model, x=2, R_s = 10 Ω, C_x= 20 fF

The 10%, 50% and 90% delays at node N_7 (Figure 4.7) are calculated first using NILT, HSPICE and the two-pole model [25] when the input signal is a unit step. Figure 4.8 (a) shows the transient responses comparing NILT, Padé order (N = 1, M = 2), HSPICE and the two-pole model and Figure 4.8 (b) shows the NILT responses of the higher order Padé approximations for x = 2, R_s = 10 Ω, and C_x = 20 fF when the input voltage is a unit step. The estimated results of 10%, 50% and 90% delay are shown in Table 4.9 for the unit step response. As observed in Tables 4.9, NILT with Padé order (N = 1, M = 2) is more accurate than the two-pole model (Table 4.9) when estimating 10%, 50% and 90% delay. For the two-pole model, the average errors for 10%, 50% and 90% delay estimation are 29.8%, 49.1% and 13.8%, respectively, while the average errors for 10%, 50% and 90% delay estimation are 19.9%, 33.8% and 10.7% for the proposed algorithm with Padé order (N = 1, M = 2). Table 4.9 also provides the 10%, 50% and 90% delay estimates of higher order Padé approximations, illustrating that Padé order (N = 3, M = 4) and higher provides better accuracy than the two-pole model. The average errors of the 10%, 50% and 90% delay estimates have dropped to 1.5%, 13.8% and 4.7% when the Padé order increases to (N = 15, M = 16) remaining numerical stability. Conversely, increasing the number of poles by using a higher order Maclaurin series to approximate the cosh and sinh terms of (2.13) following the steps of [25] results in unstable three-pole models for this example shown as Figure 4.9.

Table 4. 9: Comparison of 10%, 50% and 90% delays at node N_7 using the proposed method and two-pole model [25] for Example 2 with unit step input

x	R_s	C_x	HSPICE			Two-Pole Model [25]			Proposed Method (Padé Order N, M)											
									(1, 2)			(3, 4)			(7, 8)			(15, 16)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
2	10	20	15.52	16.69	45.66	9.50	25.50	43.00	11.68	22.88	44.75	13.73	18.58	51.62	14.72	17.14	48.17	15.26	16.46	45.31
	10	100	16.02	18.00	59.39	10.80	29.40	51.60	12.52	24.88	55.47	14.63	20.14	70.00	15.48	18.64	59.25	15.81	17.96	60.09
	30	20	15.63	46.15	119.28	10.80	42.60	125.40	12.88	39.45	137.16	14.30	24.11	122.34	15.04	19.54	101.82	15.42	44.85	114.83
	30	100	16.68	29.52	128.50	12.60	52.20	158.40	13.81	48.79	148.63	15.29	25.76	146.90	15.88	57.96	122.92	16.05	58.86	130.17
4	10	20	15.51	16.70	85.63	10.80	31.20	64.20	11.68	22.91	68.18	13.73	18.58	89.44	14.72	17.15	74.57	15.26	16.46	72.67
	10	100	15.93	18.20	94.59	11.40	34.20	75.60	12.54	24.25	81.37	14.63	20.14	104.71	15.48	18.64	84.72	15.81	17.96	81.54
	30	20	15.63	70.07	191.06	11.40	64.80	210.00	12.87	54.38	204.18	14.30	24.16	195.39	15.04	19.53	199.85	15.42	69.81	190.40
	30	100	16.12	81.94	225.11	12.00	73.80	243.60	13.84	63.57	236.92	15.29	26.13	221.03	15.88	82.50	247.27	16.05	79.14	222.54
Average Error (%)						29.8	49.1	13.8	19.9	33.8	10.7	8.8	29.9	8.4	3.8	29.8	7.8	1.5	13.8	4.7
Maximum Error (%)						38.8	87.9	25.0	24.7	65.3	20.4	11.5	68.1	17.9	5.2	96.3	14.6	3.8	99.4	15.1

To demonstrate the efficiency of the proposed algorithm, the average run times to solve the transient responses for nodes N_1 to N_7 at 500 time points are calculated using the proposed method and the two-pole model, respectively. For this example, the average run times for the two-pole model and the provided algorithm with Padé order (N = 1, M = 2) are 8.84 ms and 16.98 ms for the case when x = 2, respectively. It shows that the two-pole model needs less run time than the proposed algorithm, which is about 2 times slower than the two-pole model.

4.1.2.2 Comparing the Proposed Algorithm to PRIMA

In order to verify the accuracy and efficiency of the proposed algorithm when analyzing the time domain transient responses for the complicated interconnect structure, such as the symmetrical unbalanced tree networks, the comparison between the proposed method and PRIMA [29] is performed in this section.

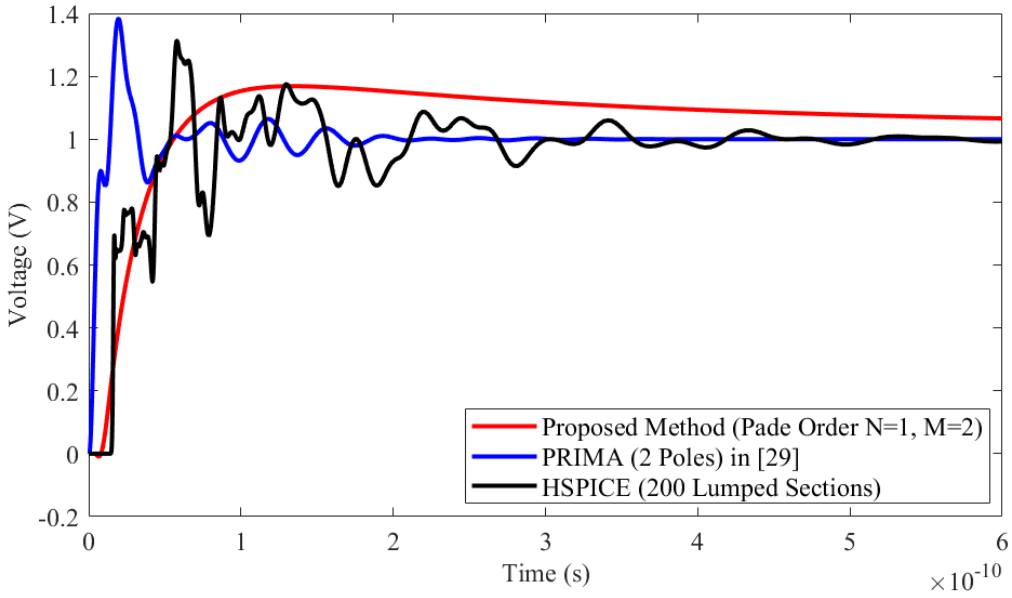
To compare with the proposed method, in which the Padé orders are selected (N = 1, M = 2), (N = 3, M = 4), (N = 7, M = 8) and (N = 15, M = 16), respectively, the approximation orders of the PRIMA are selected 2, 4, 8 and 16 accordingly. In this example, the considered symmetrical unbalanced tree interconnect is a five-port network. According to the formula (2.80), 1, 2 and 4 moments are matched when four different approximation orders are selected, respectively. In the moment matching process, approximation orders 2 and 4 generate one moment matching, approximation orders 8 and 16 generate two and four moments matching, respectively. In the application of PRIMA, each transmission line in the symmetrical unbalanced tree structure is modeled by 27 lumped *RLC* sections. Figure 4.10 (a) shows the transient responses obtained from the NILT with Padé order (N = 1, M = 2), HSPICE and the PRIMA with approximation orders 2, respectively, for x = 2, R_s = 10 **Ω**, C_x = 20 fF. Figure 4.10 (b)-(d) shows the comparison of responses of the higher order approximations between the NILT (Padé orders (N = 3, M = 4), (N = 7, M = 8) and (N = 15, M = 16), respectively) and the PRIMA (approximation orders 4, 8 and 16, respectively) for the same circuit parameters. From the Figure 4.10, the proposed algorithm can provide more accurate estimations for 10%, 50% and 90% delay compared to PRIMA with the unit step signal input, while the PRIMA is more accurate at the steady state. Once again, the

proposed algorithm can improve the accuracy of delay estimation by increasing the Padé order while keeping numerical stability. The delay results at the node N_7 are calculated using the proposed algorithm and PRIMA, and compared to those of HSPICE to verify the accuracy of the proposed algorithm (Table 4.10).

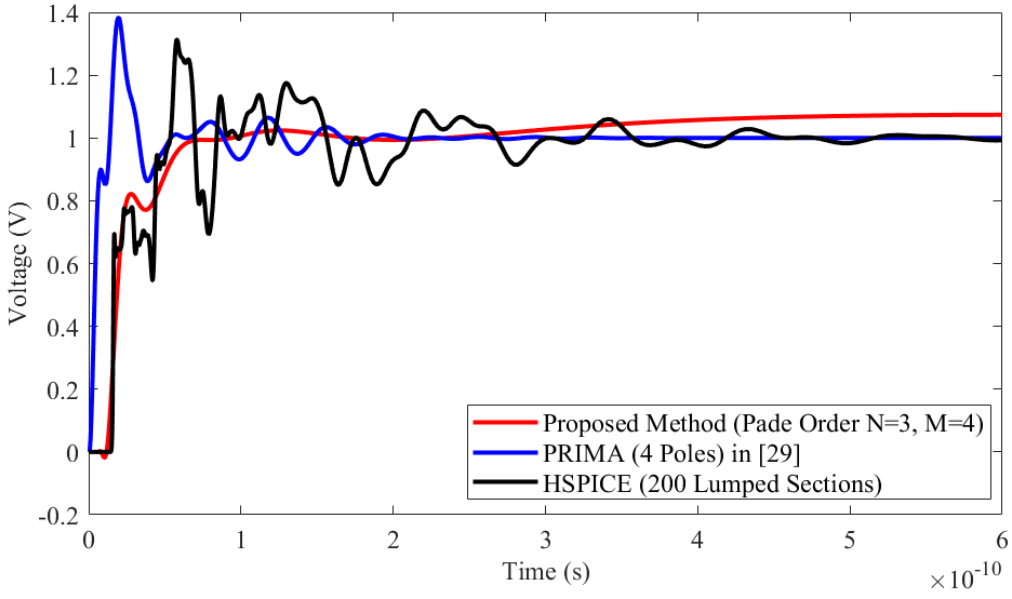
As presented in Table 4.10, the average errors for 10%, 50% and 90% delay estimations are 83.4%, 75.2% and 80.2%, respectively, for PRIMA with the approximation order of 2. By comparison, the average errors for 10%, 50% and 90% delay estimations are 19.9%, 33.8% and 10.7%, respectively, for the proposed algorithm with Padé order (N = 1, M = 2). Since the approximation order of 4 of PRIMA matches only one moment like the approximation order of 2, the accuracy of the delay estimations is not improved though the approximation order of PRIMA is increased. Conversely, the accuracies of 10%, 50% and 90% delay estimations are improved illustrated by the decreased average errors of delay estimations when the Padé order is increased from (N = 1, M = 2) to (N = 3, M = 4) (Table 4.10). For PRIMA, when the approximation order is increased to 8, the accuracies of delay estimations are improved significantly compared to the lower approximation orders, which dropped to 19.7%, 32.0% and 20.3%, respectively. Further increasing the approximation order to 16, the accuracies of the delay estimations keep unchanged comparing with approximation order of 8 though the accuracy of steady state is improved. As a comparison, the proposed algorithm improves the accuracies of delay estimations obviously when the Padé order increases. For Padé order (N = 15, M = 16), the average errors of 10%, 50% and 90% delay are 1.5%, 13.8% and 4.7%, respectively, which have been decreased greatly compared with lower Padé orders.

Performing run times comparison between the proposed method and PRIMA to illustrate the efficiency of the proposed algorithm when analyzing the complicated interconnect structure. The results of average run times to calculate the transient responses for nodes N_1 to N_7 at 500 time points using the proposed method and PRIMA are listed in Table 4.11 for the case when x = 2. As seen from Table 4.11, the proposed method has linear complexity with the order of the Padé approximation and requires less run time than PRIMA. For the same approximation order of 2, the average run times for PRIMA and the proposed method are 580.29 ms and 16.98 ms, which denotes that the proposed algorithm

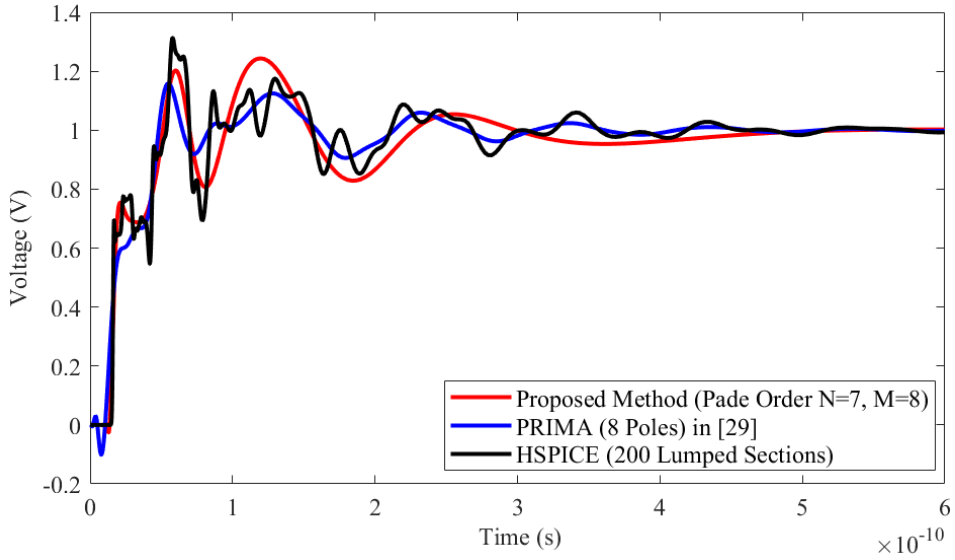
is about 34 times faster than PRIMA. Increasing the approximation order to 16 for both methods, the average run times for PRIMA and the proposed algorithm increase to 627.00 ms and 105.95 ms, respectively. It shows that the proposed algorithm is about 6 times faster than PRIMA for the highest approximation order in this example when the input signal is unit step.



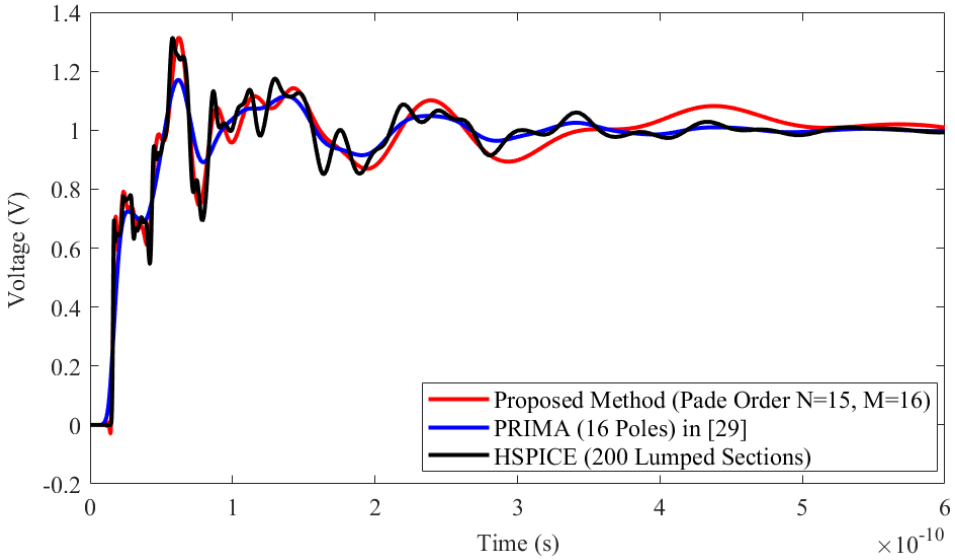
(a)



(b)



(c)



(d)

Figure 4. 10: The far end time domain response at node N_7 for Example 2 with unit step input, $x = 2$, $R_s = 10\ \Omega$, $C_x = 20$ fF (a) Comparison of proposed method Padé order ($N = 1$, $M = 2$), HSPICE and PRIAM (2 poles) [29] (b) Comparison of proposed method Padé order ($N = 3$, $M = 4$), HSPICE and PRIAM (4 poles) [29] (c) Comparison of proposed method Padé order ($N = 7$, $M = 8$), HSPICE and PRIAM (8 poles) [29] (d) Comparison of proposed method Padé order ($N = 15$, $M = 16$), HSPICE and PRIAM (16 poles) [29]

Table 4. 10: Comparison of 10%, 50% and 90% delays at node N_7 for Example 2 using the proposed method and PRIMA [29] with unit step input

x	R_s	C_x	HSPICE			PRIMA [29] (Approximation Order)						Proposed Method (Padé Order N, M)					
						2			4			(1, 2)			(3, 4)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
2	20	20	15.52	16.69	45.66	3.66	7.47	10.17	3.66	7.47	10.17	11.68	22.88	44.75	13.73	18.58	51.62
	100	100	16.02	18.00	59.39	3.33	8.13	31.74	3.33	8.13	31.74	12.52	24.88	55.47	14.63	20.14	70.00
	20	20	15.63	46.15	119.28	1.74	4.14	14.76	1.74	4.14	14.76	12.88	39.45	137.16	14.30	24.11	122.34
	100	100	16.68	29.52	128.50	3.12	7.68	40.62	3.12	7.68	40.62	13.81	48.79	148.63	15.29	25.76	146.90
4	20	20	15.51	16.70	85.63	1.59	3.42	5.22	1.59	3.42	5.22	11.68	22.91	68.18	13.73	18.58	89.44
	100	100	15.93	18.20	94.59	3.03	7.08	10.98	3.03	7.08	10.98	12.54	24.25	81.37	14.63	20.14	104.71
	20	20	15.63	70.07	191.06	1.62	3.57	5.85	1.62	3.57	5.85	12.87	54.38	204.18	14.30	24.16	195.39
	100	100	16.12	81.94	225.11	3.12	7.68	40.62	3.12	7.68	40.62	13.84	63.57	236.92	15.29	26.13	221.03
Average Error (%)						83.4	75.2	80.2	83.4	75.2	80.2	19.9	33.8	10.7	8.8	29.9	8.4
Maximum Error (%)						89.7	94.9	96.9	89.7	94.9	96.9	24.7	65.3	20.4	11.5	68.1	17.9

Table 4.10 (Cont'd.):

x	R_s	C_x	HSPICE			PRIMA [29] (Approximation Order)						Proposed Method (Padé Order N, M)					
						8			16			(7, 8)			(15, 16)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
2	20	20	15.52	16.69	45.66	17.01	26.88	32.76	17.01	26.88	32.76	14.72	17.14	48.17	15.26	16.46	45.31
	100	100	16.02	18.00	59.39	12.93	19.74	57.09	12.93	19.74	57.09	15.48	18.64	59.25	15.81	17.96	60.09
	20	20	15.63	46.15	119.28	12.39	45.03	114.09	12.39	45.03	114.09	15.04	19.54	101.82	15.42	44.85	114.83
	100	100	16.68	29.52	128.50	13.08	70.86	213.48	13.08	70.86	213.48	15.88	57.96	122.92	16.05	58.86	130.17
4	20	20	15.51	16.70	85.63	11.67	15.72	59.88	11.67	15.72	59.88	14.72	17.15	74.57	15.26	16.46	72.67
	100	100	15.93	18.20	94.59	12.69	17.22	73.77	12.69	17.22	73.77	15.48	18.64	84.72	15.81	17.96	81.54
	20	20	15.63	70.07	191.06	12.09	57.36	196.35	12.09	57.36	196.35	15.04	19.53	199.85	15.42	69.81	190.40
	100	100	16.12	81.94	225.11	13.08	70.86	213.48	13.08	70.86	213.48	15.88	82.50	247.27	16.05	79.14	222.54
Average Error (%)						19.7	32.0	20.3	19.7	32.0	20.3	3.8	29.8	7.8	1.5	13.8	4.7
Maximum Error (%)						24.8	140.0	66.1	24.8	140.0	66.1	5.2	96.3	14.6	3.8	99.4	15.1

Table 4. 11: Run Time Comparison between the Proposed Method and PRIMA [29] for Example 2 with Unit Step Input

PRIMA [29]		Proposed Method		Speed Up
Approximation Order	Run Time (ms)	Padé Order	Run Time (ms)	
2	580.29	(1, 2)	16.98	34
4	580.29	(3, 4)	31.07	19
8	600.02	(7, 8)	56.60	11
16	627.00	(15, 16)	105.95	6

4.1.3 Example 3- Unsymmetrical Distributed RLC Tree

4.1.3.1 Comparing the Proposed Algorithm to Two-Pole Model

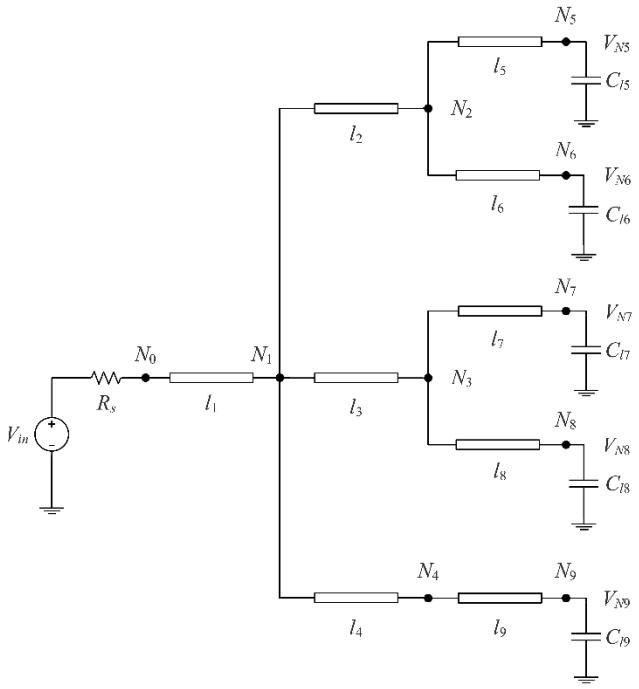


Figure 4. 11: Unsymmetrical distributed *RLC* tree

This example analyzes the unsymmetrical tree network of Figure 4.11. The per-unit-length parameters are the same values as used in Example 2 and the normalized wire lengths and

load capacitances are listed in Table 4.12 and Table 4.13, where l_x and C_x are the normalized reference length and capacitance, respectively. The variable l_x is set 0.2 and 1, respectively. The 10%, 50% and 90% delays at node N_7 (Figure 4.11) are calculated using NILT, HSPICE, the two-pole model [25] and PRIMA [29] for various line lengths, resistive loads and capacitive loads.

Table 4. 12: Interconnect lengths normalized to l_x used for Example 3

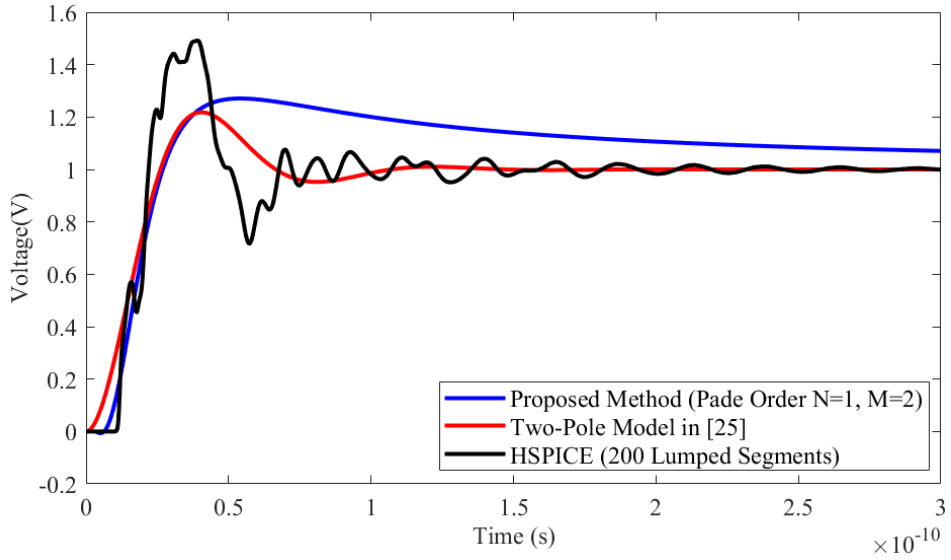
Index	l_1	l_2	l_3	l_4	l_5	l_6	l_7	l_8	l_9
Length	0.5	1	2	0.5	1	0.5	2	1	1

Table 4. 13: Load capacitances normalized to C_x used for Example 3

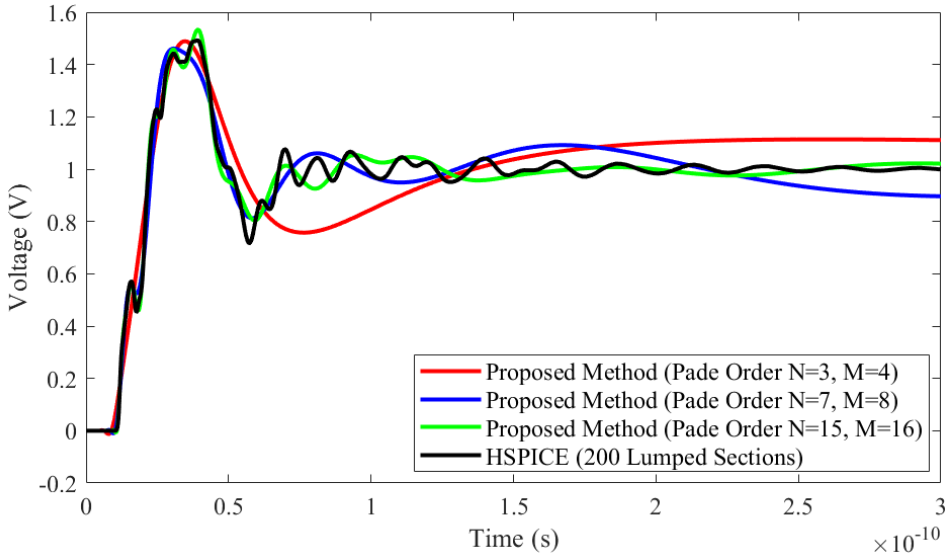
Index	C_{l5}	C_{l6}	C_{l7}	C_{l8}	C_{l9}
Capacitance	2	0.5	2	5	1

Table 4.14 shows the results for the unit step response obtained from HSPICE, the two-pole model and the proposed algorithm, respectively. Figure 4.12 (a) shows the transient responses comparing NILT, Padé order (N = 1, M = 2), HSPICE and the two-pole model for l_x= 0.2 mm, R_s = 10 Ω, and C_x = 20 fF when the input voltage is a unit step. Figure 4.12 (b) shows the NILT responses of the higher order Padé approximations. Once again, Tables 4.14 shows that both NILT and the two-pole model become less accurate for the unit step input due to the infinitely small rising time. By comparison, NILT Padé order (N = 1, M = 2) is more accurate than the two-pole model (Table 4.14) for the 10%, 50% and 90% delays for the unit step response. Also increasing the Padé order of the NILT approximation provides better delay estimates when compared to the two-pole model. When the Padé order of the proposed method is increased from (N = 1, M = 2) to (N = 15, M = 6), the average errors of the 10%, 50% and 90% delays are dropped to 0.41%, 2.6% and 1.5% from 10.9%, 6.1% and 24.0%, respectively. Accordingly, the maximum errors of the delays are decreased to 1.4%, 10.5% and 3.8% from 17.1%, 16.4% and 59.2%, respectively.

The average run times to solve nodes N_1 to N_9 at 500 time points using NILT and the two-pole model are calculated and compared for the case when l_x = 0.2 mm. In this example



(a)



(b)

Figure 4. 12: The far end time domain response at node N_7 for Example 3 with unit step input, l_x = 0.2 mm, R_s = 10 Ω, C_x= 20 fF (a) Comparison of proposed method Padé order (N = 1, M = 2), HSPICE and two-pole model [25] (b) Comparison of proposed method Padé orders (N = 3, M = 4), (N = 7, M = 8) and (N = 15, M = 16), respectively, against HSPICE

the average run times are 12.22 ms and 14.58 ms for the two-pole model and the proposed algorithm with Padé order of (N = 1, M = 2), respectively. Once again, the two-pole model

Table 4. 14: Comparison of 10%, 50% and 90% delays at node N_7 using the proposed algorithm and the two-pole model [25] for Example 3 with unit step input

l_x	R_s	C_x	HSPICE			Two-Pole Model [25]			Proposed Method (Padé Order N, M)											
									(1, 2)			(3, 4)			(7, 8)			(15, 16)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
(mm)	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
0.2	10	20	11.61	14.36	21.81	5.70	14.70	23.40	9.87	16.72	24.20	10.86	15.26	21.29	11.46	14.32	22.13	11.54	14.59	21.30
	10	500	25.22	61.51	105.92	18.60	58.50	134.4	23.52	62.69	128.79	22.58	60.19	109.87	25.70	59.79	106.16	25.25	60.70	107.77
	30	20	11.81	23.57	36.97	6.60	22.20	57.30	11.25	23.25	58.86	11.57	21.06	65.79	11.78	21.96	64.55	11.76	21.09	36.75
	30	100	13.60	37.07	106.71	10.80	40.80	117.60	15.76	40.96	101.20	14.56	37.30	138.89	13.67	37.30	106.09	13.58	37.23	106.95
1	10	20	56.55	79.43	101.10	26.65	71.50	119.60	46.86	80.01	121.84	52.60	76.18	99.83	54.78	71.05	101.29	55.77	75.38	97.30
	10	100	58.33	99.53	116.38	31.20	85.15	146.25	51.83	94.01	149.33	56.00	88.61	122.73	58.18	98.55	120.53	58.28	99.33	115.28
	20	20	56.44	90.09	135.07	29.25	87.10	178.10	49.88	94.32	169.35	54.26	85.45	126.54	55.49	91.17	126.20	56.18	91.53	132.84
	20	100	58.89	105.00	175.93	35.10	105.30	225.55	55.79	112.77	213.06	58.07	102.86	164.83	59.12	109.01	176.66	58.94	105.13	177.66
Average Error (%)						41.2	6.4	25.4	10.9	6.1	24.0	5.3	5.2	16.7	1.2	3.4	11.0	0.41	2.6	1.5
Maximum Error (%)						52.9	14.4	55.0	17.1	16.4	59.2	10.5	11.0	78.0	3.1	10.6	74.6	1.4	10.5	3.8

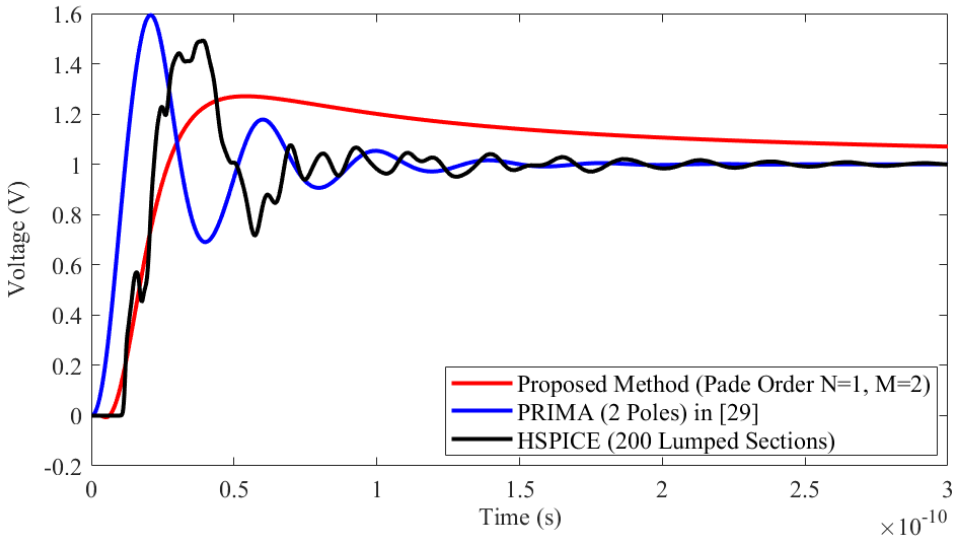
requires less time than the proposed algorithm, which is slightly slow than the two-pole model. However, the two-pole model still suffers the instabiltiy and numerically problems as analyzed in previous sections due to the fact that large inductance of transmission line results in the instable second-order approximation of moment matching. As well, an unstable three-pole model for this example will be generated when increasing the number of poles by using a higher order Maclaurin series to approximate the cosh and sinh terms of (2.13) following the steps of [25].

4.1.3.2 Comparing the Proposed Algorithm to PRIMA

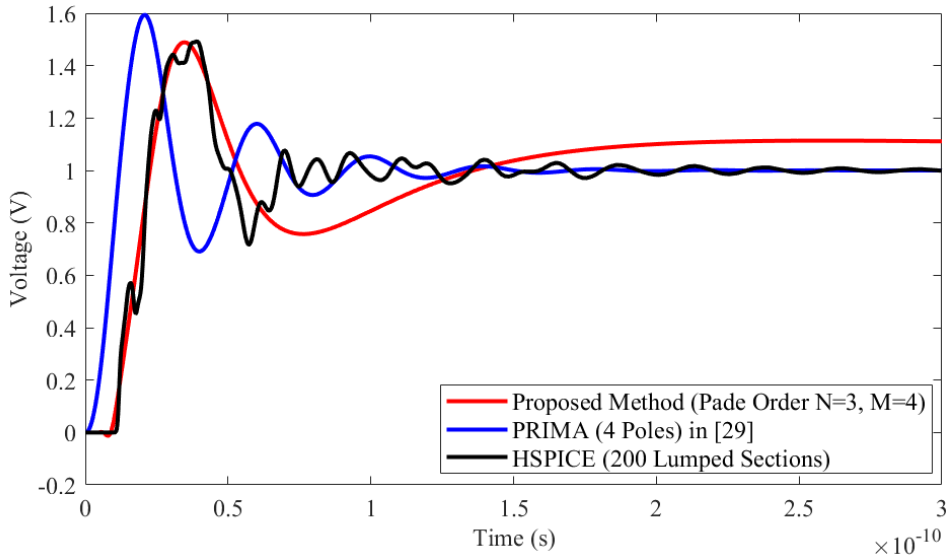
In this section, PRIMA [29] as an advanced moment matching based method, which can provide desired accuracy of *RLC* reduced model for any approximation order without numerical issues is considered and compared to the proposed algorithm to verify both the accuracy and efficiency of the proposed method. In this example, the given unsymmetrical tree interconnect is a six-port network, the approximation orders of 2, 4, 8 and 16 for PRIMA are considered to keep consistent with the Padé orders of (N = 1, M = 2), (N = 3, M = 4), (N = 7, M = 8) and (N = 15, M = 16) for the proposed algorithm. According to formula (2.80), approximation orders 2 and 4 generates one moment matching, approximation orders 8 and 16 generate two and three moments matching, respectively. When using PRIMA to analyze the given complicated tree structure, each transmission line is modeled by 27 lumped *RLC* sections. Figure 4.13 (a) shows the transient responses obtained from the NILT with Padé order (N = 1, M = 2), HSPICE and the PRIMA with approximation order of 2, respectively, for l_x = 0.2 mm, R_s = 10 Ω, C_x= 20 fF. Figure 4.13 (b)-(d) shows the comparison of responses of the higher order approximations between the NILT (Padé orders (N = 3, M = 4), (N = 7, M = 8) and (N = 15, M = 16), respectively) and the PRIMA (approximation orders 4, 8 and 16, respectively) for the same circuit parameters. From the Figure 4.13, the proposed algorithm provides more accurate estimations for 10%, 50% and 90% delay compared to PRIMA with the unit step signal input, while the PRIMA is more accurate at the steady state. Once again, the proposed algorithm can improve the accuracy of delay estimation by increasing the Padé order while keeping numerical stability. The delay results at the node N_7 are calculated using the

proposed algorithm and PRIMA, respectively and compared to those of HSPICE to verify the accuracy of the proposed algorithm (Table 4.15).

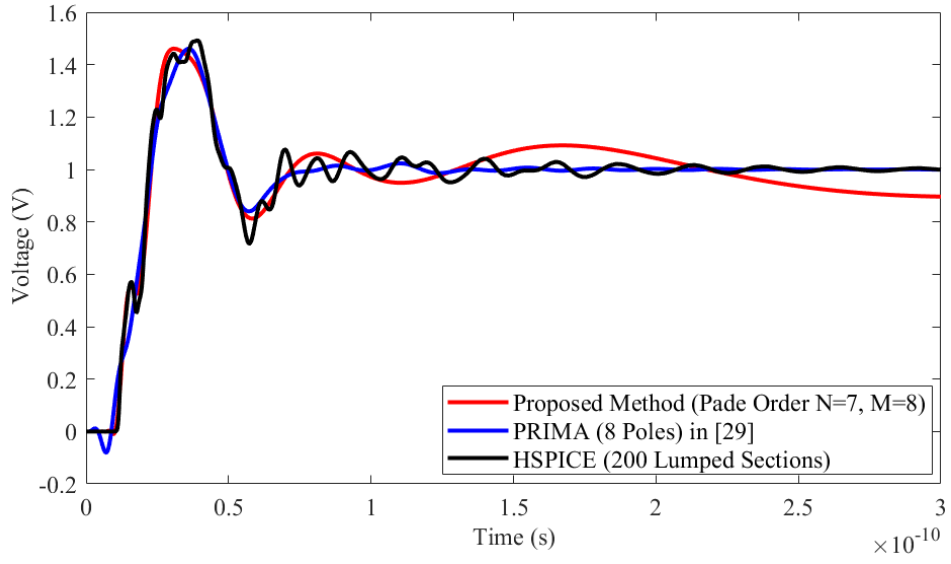
As discussed above, approximation orders of 2 and 4 for PRIMA generate the same results of 10%, 50% and 90% delay since they both can match only one moment (Table 4.15). For the proposed algorithm, the average error of the 10% delay estimation is dropped to 5.3% from 10.9%, the average error of the 50% delay is dropped to 5.2% from 6.1%, and the average error of the 90% delay is dropped to 16.7% from 24.0%, respectively,



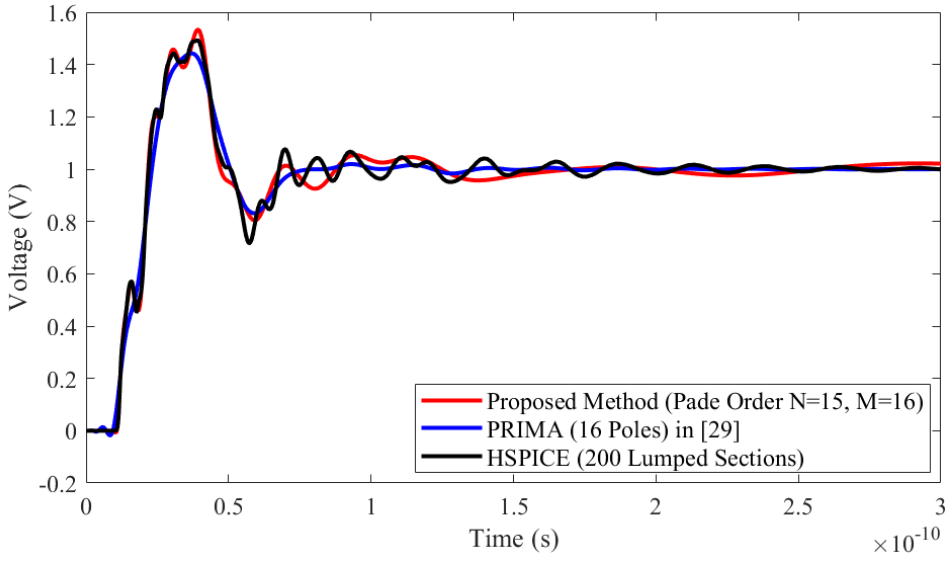
(a)



(b)



(c)



(d)

Figure 4. 13: The far end time domain response at node N_7 for Example 3 with unit step input, l_x = 0.2 mm, R_s = 10 Ω, C_x= 20 fF (a) Comparison of proposed method Padé order (N = 1, M = 2), HSPICE and PRIAM (2 poles) [29] (b) Comparison of proposed method Padé order (N = 3, M = 4), HSPICE and PRIAM (4 poles) [29] (c) Comparison of proposed method Padé order (N = 7, M = 8), HSPICE and PRIAM (8 poles) [29] (d) Comparison of proposed method Padé order (N = 15, M = 16), HSPICE and PRIAM (16 poles) [29]

Table 4. 15: Comparisons of 10%, 50% and 90% delays at node N_7 for Example 3 using the proposed method and PRIMA [29] with unit step input

l_x	R_s	C_x	HSPICE			PRIMA [29] (Approximation Order)						Proposed Method (Padé Order N, M)					
						2			4			(1, 2)			(3, 4)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
(mm)	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
0.2	10	20	11.61	14.36	21.81	3.48	7.79	11.06	3.48	7.79	11.06	9.87	16.72	24.20	10.86	15.26	21.29
	10	500	25.22	61.51	105.92	20.90	55.82	98.36	20.90	55.82	98.36	23.52	62.69	128.79	22.58	60.19	109.87
	30	20	11.81	23.57	36.97	3.86	9.27	14.10	3.86	9.27	14.10	11.25	23.25	58.86	11.57	21.06	65.79
	30	100	13.60	37.07	106.71	10.10	27.35	77.81	10.10	27.35	77.81	15.76	40.96	101.20	14.56	37.30	138.89
1	10	20	56.55	79.43	101.10	7.61	16.77	23.60	7.61	16.77	23.60	46.86	80.01	121.84	52.60	76.18	99.83
	10	100	58.33	99.53	116.38	17.26	40.17	58.73	17.26	40.17	58.73	51.83	94.01	149.33	56.00	88.61	122.73
	20	20	56.44	90.09	135.07	7.77	17.52	24.99	7.77	17.52	24.99	49.88	94.32	169.35	54.26	85.45	126.54
	20	100	58.89	105.00	175.93	18.20	44.04	67.11	18.20	44.04	67.11	55.79	112.77	213.06	58.07	102.86	164.83
Average Error (%)						61.6	52.4	51.9	61.6	52.4	51.9	10.9	6.1	24.0	5.3	5.2	16.7
Maximum Error (%)						86.5	80.6	81.5	86.5	80.6	81.5	17.1	16.4	59.2	10.5	11.0	78.0

Table 4.15 (Cont'd.):

l_x	R_s	C_x	HSPICE			PRIMA [29] (Approximation Order)						Proposed Method (Padé Order N, M)					
						8			16			(7, 8)			(15, 16)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
(mm)	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
0.2	10	20	11.61	14.36	21.81	9.93	16.91	21.75	9.93	16.91	21.75	11.46	14.32	22.13	11.54	14.59	21.30
	10	500	25.22	61.51	105.92	24.53	60.21	108.06	24.53	60.21	108.06	25.70	59.79	106.16	25.25	60.70	107.77
	30	20	11.81	23.57	36.97	10.25	21.48	63.27	10.25	21.48	63.27	11.78	21.96	64.55	11.76	21.09	36.75
	30	100	13.60	37.07	106.71	13.14	37.82	108.39	13.14	37.82	108.39	13.67	37.30	106.09	13.58	37.23	106.95
1	10	20	56.55	79.43	101.10	49.79	75.79	100.10	49.79	75.79	100.10	54.78	71.05	101.29	55.77	75.38	97.30
	10	100	58.33	99.53	116.38	49.47	89.90	118.72	49.47	89.90	118.72	58.18	98.55	120.53	58.28	99.33	115.28
	20	20	56.44	90.09	135.07	50.86	86.45	133.09	50.86	86.45	133.09	55.49	91.17	126.20	56.18	91.53	132.84
	20	100	58.89	105.00	175.93	49.92	106.93	169.91	49.92	106.93	169.91	59.12	109.01	176.66	58.94	105.13	177.66
Average Error (%)						10.8	6.4	10.4	10.8	6.4	10.4	1.2	3.4	11.0	0.41	2.6	1.5
Maximum Error (%)						15.2	17.8	71.1	15.2	17.8	71.1	3.1	10.6	74.6	1.4	10.5	3.8

when the Padé order is increased from (N = 1, M = 2) to (N = 3, M = 4). For PRIMA, when the approximation order is increased to 8, the accuracies of delay estimations are improved significantly compared to the lower approximation orders, which are dropped to 10.8%, 6.4% and 10.4%, respectively. Further increasing the approximation order to 16 for PRIMA, the accuracies of the delay estimations keep unchanged comparing with approximation order of 8 though the accuracy of steady state is improved. As a comparison, the proposed algorithm improves the accuracies of delay estimations obviously when the Padé order increases. For Padé order (N = 15, M = 16), the average errors of 10%, 50% and 90% delay are 0.41%, 2.6% and 1.5%, respectively, which have been decreased greatly compared with lower Padé orders.

Table 4. 16: Run time comparison between the proposed method and PRIMA [29] for Example 3 with unit step input

PRIMA [29]		Proposed Method		Speed Up
Approximation Order	Run Time (ms)	Padé Order	Run Time (ms)	
2	368.84	(1, 2)	14.53	25
4	368.84	(3, 4)	20.83	18
8	437.37	(7, 8)	35.49	12
16	447.05	(15, 16)	65.05	7

In order to demonstrate the efficiency of the proposed algorithm run times used to solve nodes N_1 to N_9 at 500 time points using both the proposed algorithm and PRIMA are calculated and compared. The results of average run times are listed in Table 4.16 for the case when l_x = 0.2 mm. As seen from Table 4.16, the proposed method has linear complexity with the order of the Padé approximation order and requires less run time than PRIMA. For the same approximation order of 2, the average run times for PRIMA and the proposed method are 368.84 ms and 14.53 ms, which shows that the proposed algorithm is about 25 times faster than PRIMA. Increasing the approximation order to 16 for both methods, the average run times for PRIMA and the proposed algorithm increase to 447.05 ms and 65.05 ms, respectively. It shows that the proposed algorithm is about 7 times faster than PRIMA for the highest approximation order in this example when the input signal is

unit step.

4.1.4 Summary of the Results

For the unit step input, the simple single transmission line, the complicated symmetrical unbalanced and unsymmetrical interconnect tree structures are analyzed to demonstrate the proposed method with respect to the accuracy and efficiency. The proposed algorithm can provide more accurate 10%, 50% and 90% delay estimations and the accuracy of the delay estimates increases as the Padé order of the proposed algorithm increases when comparing to the two-pole model [25], which provides more accurate estimation at steady state. Though the two-pole model [25] requires less run time than the proposed algorithm when calculating the time domain transient response at the given node, the two-pole model [25], which is based on the second-order approximation of moment matching technique, experiences numerically stable problem because the large inductance value of the transmission line results in minus coefficient when matching moments. Unlike the two-pole model [25], the proposed algorithm uses a numerical inversion of the Laplace transform (NILT) to convert the frequency solution to a time function, suitable for transient analysis. Since the integration formula of NILT is numerically stable for higher order approximations, the developed algorithm provides a mechanism to increase the accuracy of the delay estimates without stability and numerical problems such as suffered by the two-pole model.

In order to further verify the accuracy and efficiency of the proposed algorithm, the advanced moment matching based PRIMA [29], which has different approximation orders and is known as to provide accurate RLC reduced models for any approximation order without numerical issues is considered and compared to the proposed algorithm. By comparison, the proposed method is capable of providing more accurate estimations at the early stage than PRIMA [29], while PRIMA [29] can provide more accurate estimations at the steady state than the proposed algorithm. As well the proposed algorithm can improve the accuracy of the delay estimates significantly with the Padé order increases, while PRIMA [29] keeps the accuracy of the delay estimates unimproved when the approximation order reaches a level. Higher approximation order of PRIMA can only

improve the accuracy of the steady state and does not make any contribution to the accuracy improvement at early stage. For PRIMA [29], a number of lumped *RLC* sections is required to characterize a transmission line. More number of lumped *RLC* sections are needed when more moments are matched to improve the accuracy of PRIMA [29]. Large number of lumped RLC sections produces large size of matrices used in PRIMA [29], which results in expensive computation cost. For the proposed algorithm, which is applied based on the Y-parameter of the given interconnect structure corresponding to the small size matrix when compared to the large size matrix of PRIMA [29]. The fact of matrix size determines that the proposed method is more efficient than PRIMA [29].

4.2 Selecting Unit Ramp Signal Input

For the unit step signal, its rise time is very sharp in time domain which means it contains so many high frequency components. All the two-pole model [25], PRIMA [29] and the proposed algorithm might not always be accurate enough in capturing the high frequency components when the unit step signal is applied to the network as excitation. As a result, the average errors obtained in calculating 10%, 50% and 90% delay are large for unit step input. When the rise time of the input signal increases and the input signal becomes ramp as compared to the unit step input, it contains less high frequency components. Consequently, the two-pole model [25], PRIMA [29] and the proposed algorithm will provide better accuracy in estimating 10%, 50% and 90% delays for ramp signal input. In this section, the unit rising ramp input is considered for single line, symmetrical unbalanced tree structure and unsymmetrical tree structure, respectively, to demonstrate the advantage of the proposed algorithm over the two-pole model [25] and PRIMA [29] in respect of both accuracy and efficiency.

4.2.1 Example 1- Single Line Interconnect

4.2.1.1 Comparing the Proposed Algorithm to Two-Pole Model

The interconnect circuit of Figure 4.1 is analyzed using a unit rising ramp input. The per-unit-length parameters are obtained from [40] and are listed in Table 4.17. The line length is 0.2 cm and the rise time of the ramp input is set to 0.1 ns and to 0.025 ns. The 10%, 50% and 90% delays at node N_1 are calculated using NILT, HSPICE, the two-pole model [25]

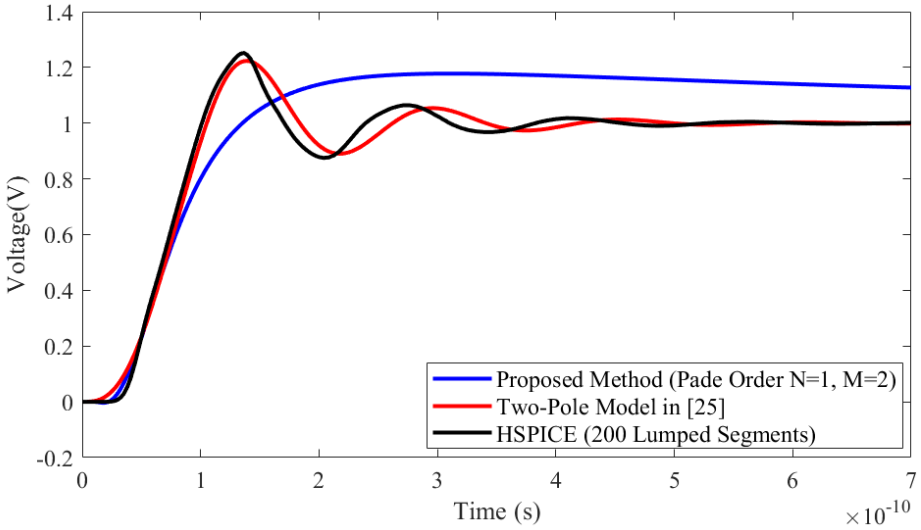
for various resistive and capacitive loads and the results are shown in Table 4.18 for the 0.1ns rise time and in Table 4.19 for the 0.025 ns rise time. Figure 4.14 (a) and Figure 4.15 (a) shows the transient response comparing NILT Padé order (N = 1, M = 2), HSPICE and the two-pole model [25] for line width = 2 μm, R_s = 20 Ω, and C_l = 10 fF when the rise times are 0.1 ns and 0.025 ns, respectively. Figure 4.14 (b) and Figure 4.15 (b) shows the corresponding higher order Padé responses when the rise times are 0.1 ns and 0.025 ns, respectively. This example illustrates that as the rise time decreases both NILT and the two-pole model become less accurate. For this example, Padé order (N = 1, M = 2) provides better estimates for the 10% delay, while the two-pole model provided better estimates for the 50% and 90% delays.

Table 4. 17: Per-unit-length parameters used for Example 1

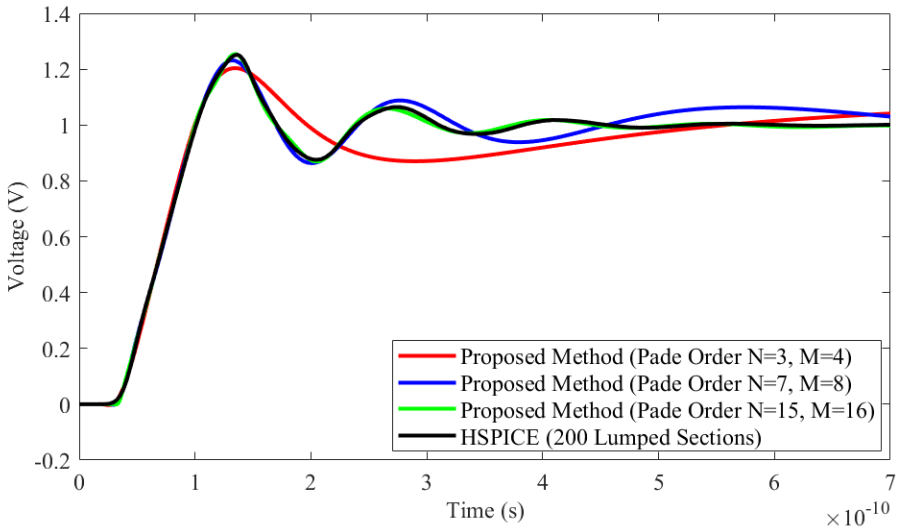
Line Width (μm)	R (Ω/cm)	L (nH/cm)	C (pF/cm)
2	88.29	15.38	1.8
6	35.50	13.60	3.3
10	22.00	12.60	4.9

However, the proposed method can increase the accuracy of the time response by increasing the order of the Padé approximation. From Table 4.18, the average errors of the 10%, 50% and 90% delay estimates generated by Padé order (N = 15, M = 16) are 0.27%, 0.066% and 0.14% for the rise time of 0.1 ns. From Table 4.19, Padé order (N = 15, M = 16) results in average errors of 0.48%, 0.29%, and 0.63% for the 10%, 50% and 90% delay estimates, respectively for the rise time of 0.025 ns. The average run times to solve for node N_1 at 500 time points using both the proposed algorithm and the two-pole model are compared. For the ramp input with rise time of 0.1 ns, the average run times are 2.35 ms and 4.09 ms for the two-pole model and the proposed algorithm with Padé order (N = 1, M = 2), respectively. For the 0.025 ns rise time scenario, the average run times are 2.38 ms and 3.69 ms for the two-pole model and the proposed algorithm with Padé order (N = 1, M = 2), respectively. The results of the average run times show that the two-pole model needs less run time than the proposed algorithm, which is about 2 times slower than the two-pole model. Though the two-pole model is faster than the proposed algorithm for the ramp signal input, it still experiences numerically stable problem due to the large inductance value of

the transmission line as it is discussed in previous examples.

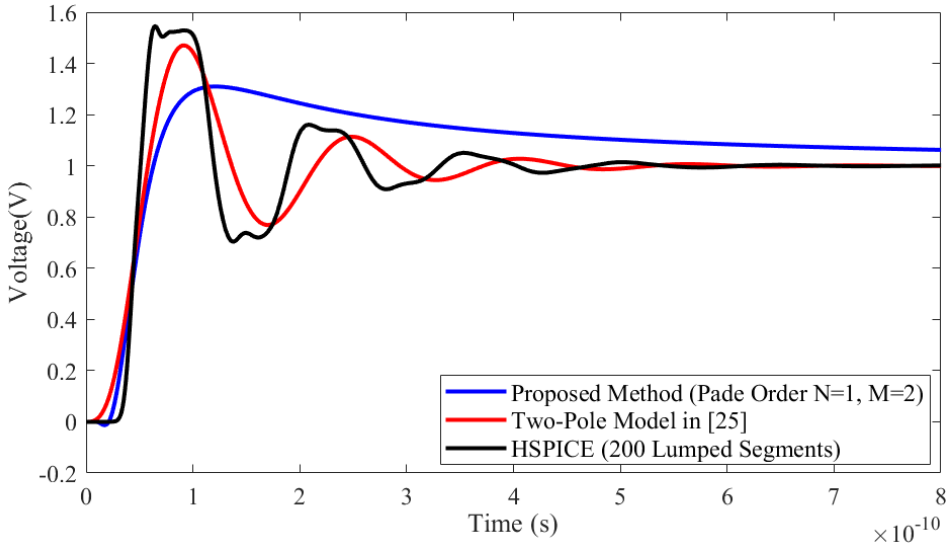


(a)

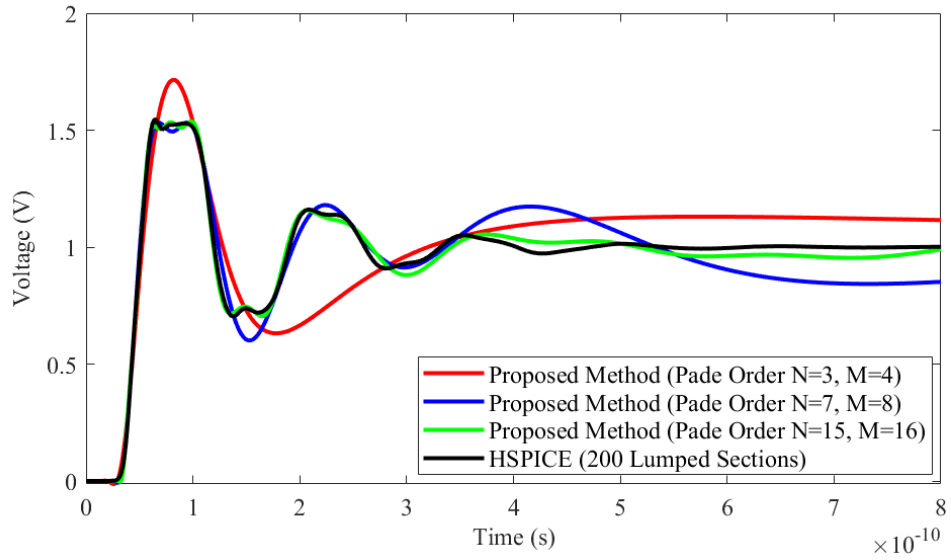


(b)

Figure 4. 14: The far end time domain response at N_1 for Example 1. Line length = 0.2 cm, line width = 2 µm, per-unit-length parameters (R = 88.29/Ω/cm, L = 15.38 nH/cm, C = 1.8 pF/cm), R_s = 20 Ω, C_l = 10 fF, the input is a ramp signal with rise time of 0.1 ns (a) Comparisons among the proposed method with Padé order (N = 1, M = 2), HSPICE and two-pole model in [25] (b) The time domain response analysis using the proposed method for Padé orders (N = 3, M = 4), (N = 7, M = 8) and (N = 15, M = 16), respectively, against HSPICE



(a)



(b)

Figure 4. 15: The far end time domain response at N_1 for Example 1. Line length = 0.2 cm, line width = 2 µm, per-unit-length parameters (R = 88.29/Ω/cm, L = 15.38 nH/cm, C = 1.8 pF/cm), R_s = 20 Ω, C_l = 10 fF, the input is a ramp signal with rise time of 0.025 ns (a) Comparisons among the proposed method with Padé order (N = 1, M = 2), HSPICE and two-pole model in [25] (b) The time domain response analysis using the proposed method for Padé orders (N = 3, M = 4), (N = 7, M = 8) and (N = 15, M = 16), respectively, against HSPICE

Table 4. 18: Comparisons of 10%, 50% and 90% delays at node N_1 using the proposed algorithm and two-pole model [25] for Example 1 when the rise time is 0.1 ns and line length is 0.2 cm

Width	R_s	C_l	HSPICE			Two-Pole Model [25]			Proposed Method (Padé Order N, M)											
									(1, 2)			(3, 4)			(7, 8)			(15, 16)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
(μm)	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
2	20	10	41.25	67.48	93.76	36.40	70.70	98.00	38.70	70.58	114.54	41.62	67.17	92.14	41.22	67.76	93.19	40.79	67.38	93.52
	50	50	46.14	79.76	112.23	40.50	82.50	117.00	44.22	83.74	140.41	46.85	79.21	111.42	46.17	80.05	112.03	46.13	79.76	112.40
	100	100	52.90	98.49	144.17	47.25	100.40	157.06	51.62	104.72	188.28	53.63	97.14	149.35	52.77	98.58	145.48	52.86	98.54	144.35
6	20	10	50.17	77.64	104.85	41.75	80.50	109.00	46.20	80.51	125.76	50.53	77.90	102.34	50.05	77.67	104.88	50.14	77.64	104.91
	50	50	56.07	92.05	128.36	46.59	94.79	136.47	52.08	97.55	165.05	55.82	91.45	128.14	55.59	92.51	127.55	56.02	92.04	128.45
	100	100	62.71	115.06	220.96	54.41	117.90	216.04	60.37	125.83	238.38	63.13	113.27	200.26	61.13	115.13	227.25	62.52	115.09	220.96
10	20	10	58.02	86.42	115.05	46.80	89.40	120.60	52.42	89.76	138.68	57.92	87.19	112.66	58.61	86.41	115.58	57.93	86.55	115.08
	50	50	62.85	103.60	144.29	52.01	106.42	161.07	58.97	111.24	192.71	63.64	102.71	146.92	63.00	104.20	144.03	62.48	103.55	143.70
	100	100	70.27	130.91	283.37	61.20	137.40	289.20	68.36	147.97	295.69	71.69	129.76	295.11	70.15	130.61	284.82	70.23	131.12	283.85
Average Error (%)						14.6	3.4	5.4	5.6	6.6	21.4	1.0	0.85	2.9	0.60	0.25	0.71	0.27	0.066	0.14
Maximum Error (%)						19.3	5.0	11.6	9.7	13.0	33.6	2.0	1.6	9.4	2.5	0.58	2.8	1.1	0.16	0.41

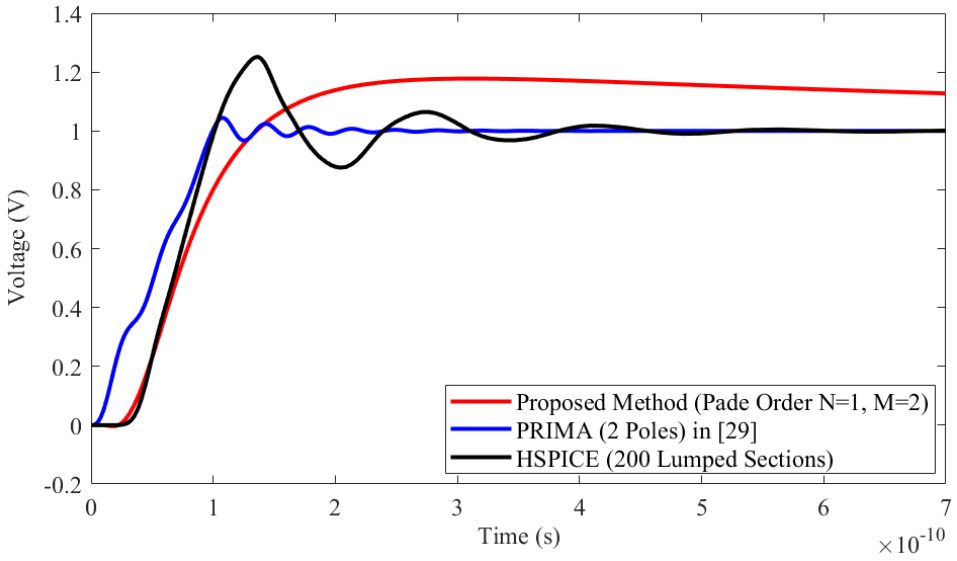
Table 4. 19: Comparisons of 10%, 50% and 90% delays at node N_1 using the proposed algorithm and two-pole model [25] for Example 1 when the rise time is 0.025 ns and line length is 0.2 cm

Width	R_s	C_l	HSPICE			Two-Pole Model [25]			Proposed Method (Padé Order N, M)											
									(1, 2)			(3, 4)			(7, 8)			(15, 16)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
(μm)	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
2	20	10	36.03	42.74	49.14	22.20	40.20	54.60	27.96	42.16	57.59	34.18	43.13	50.47	35.68	42.81	48.76	35.93	42.53	49.26
	50	50	38.66	48.34	57.10	25.20	48.00	68.40	31.26	50.48	75.47	37.40	48.96	59.76	38.62	48.26	56.78	38.57	48.36	56.84
	100	100	41.84	56.18	97.53	27.28	59.62	108.48	35.53	64.64	114.77	40.99	57.65	80.94	41.84	55.84	100.94	41.63	55.99	97.15
6	20	10	45.40	51.99	58.41	25.80	48.00	66.60	33.96	51.22	70.34	41.99	52.15	60.77	44.38	52.11	58.10	44.72	51.74	58.12
	50	50	47.32	57.36	67.18	27.60	56.70	88.80	37.43	61.76	96.90	45.05	58.43	72.13	47.24	57.27	66.74	47.32	57.35	66.62
	100	100	50.18	65.69	167.88	31.20	76.20	172.20	42.30	81.54	162.65	48.86	68.73	244.22	50.57	65.28	163.32	50.31	65.40	165.34
10	20	10	52.50	59.66	66.37	28.20	55.20	77.40	38.99	59.28	82.56	48.32	59.78	69.83	51.57	59.10	66.16	52.14	59.79	66.70
	50	50	54.56	65.21	76.79	30.75	66.75	114.00	42.87	72.89	121.20	51.45	67.03	84.82	54.41	65.16	76.61	54.58	65.21	76.07
	100	100	57.15	74.56	272.36	35.40	96.60	248.80	48.52	100.13	221.13	55.56	79.83	315.29	57.87	74.22	304.08	57.61	74.12	271.50
Average Error (%)						39.9	8.6	18.3	20.1	11.2	26.2	4.6	2.4	12.5	0.84	0.38	2.3	0.48	0.29	0.63
Maximum Error (%)						46.3	29.6	48.5	25.7	34.3	57.8	8.0	7.1	45.5	2.2	0.94	11.6	1.5	0.59	1.5

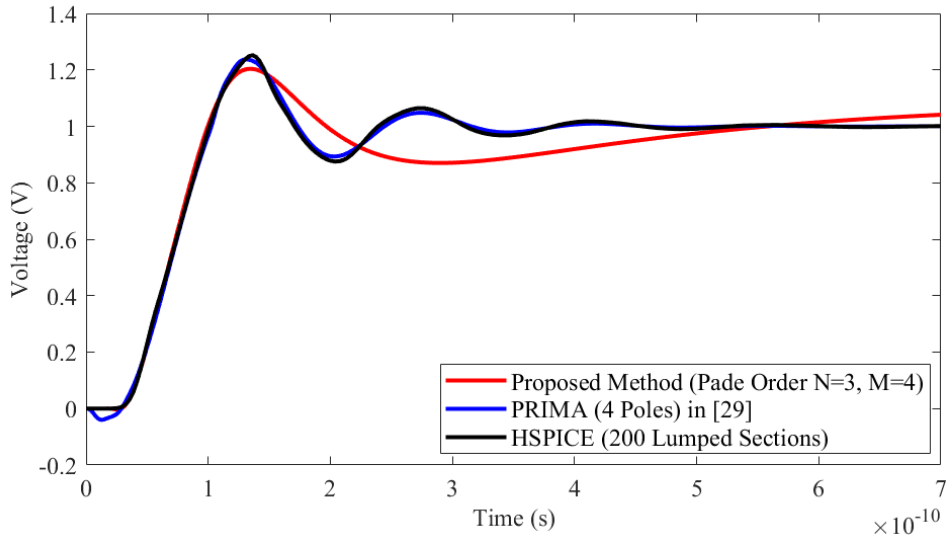
4.2.1.2 Comparing the Proposed Algorithm to PRIMA

To demonstrate the accuracy and efficiency of the proposed algorithm, PRIMA [29] is considered and compared to the proposed algorithm in this section when the input signal is ramp with rise time of 0.1 ns and 0.025 ns, respectively. For the two-port single line structure, the approximation orders of the PRIMA are selected 2, 4, 8 and 16 to correspond to the Padé orders ((N = 1, M = 2), (N = 3, M = 4), (N = 7, M = 8) and (N = 15, M = 16)) for the proposed algorithm. The calculations of 10%, 50% and 90% delays at node N_1 using the NILT, HSPICE and PRIMA [29] for various resistive and capacitive loads are listed in Tables 4.20 and 4.21. Figure 4.16 (a) and Figure 4.17 (a) show the transient responses obtained from the NILT with Padé orders (N = 1, M = 2), HSPICE and the PRIMA with approximation order 2, respectively, for line length = 0.2 cm, line width = 2 μm, per-unit-length parameters (R = 88.29/Ω/cm, L = 15.38 nH/cm, C = 1.8 pF/cm), R_s = 20 Ω, C_l = 10 fF when the rise time is 0.1 ns and 0.025 ns for the ramp signal input. Figure 4.16 (b) – 4.16 (d) and Figure 4.17 (b) – 4.17 (d) show the transient responses obtained from the NILT, HSPICE and PRIMA with the higher approximation orders, respectively. It is observed from Figure 4.16 and 4.17 that the NILT algorithm is accurate at time equal to zero and less accurate as time increases, while PRIMA is more accurate at steady state and less accurate in early time. For 0.1 ns rise time, the NILT with Padé order (N = 1, M = 2) has average errors of 5.6%, 6.6%, and 21.4% for the 10%, 50% and 90% delay estimates, respectively, while PRIMA has average errors of 63.0%, 41.7% and 29.5%, respectively when approximation order of 2 is considered (Table 4.20). Tables 4.20 and 4.21 also provides 10%, 50% and 90% delay estimates for both the NILT with higher Padé orders of (N = 3, M = 4), (N = 7, M = 8) and (N = 15, M = 16), and PRIMA with higher approximation orders of 4, 8 and 16. As illustrated in Tables 4.20 and 4.21 that as the Padé approximation order of the NILT increases, the accuracy of the delay estimates is improved greatly. As seen from Table 4.20, Padé order (N = 15, M = 16) for the NILT has the lowest average errors of 0.27%, 0.066%, and 0.14% for the 10%, 50% and 90% delay estimates. As a comparison, the accuracy of 10%, 50% and 90% delay estimates of PRIMA is improved significantly when the approximation order increases from 2 to 4 with the average error dropped to 7.1% from 63.0% for 10% delay, average error dropped to 1.1% from 47.7%

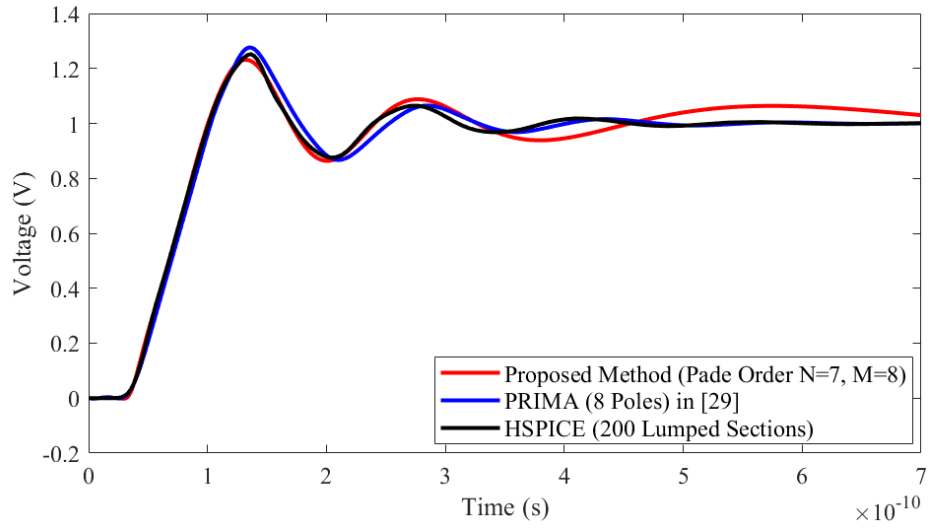
for 50% delay, and average error dropped to 2.7% from 29.5% for 90% delay, respectively. However, the accuracies of 10%, 50% and 90% delay estimations are kept unimproved when the approximation order of PRIMA is increased from 8 to 16 though the accuracy of steady state is improved as the increase of approximation orders. Both of the approximation orders 8 and 16 have the same average errors of 0.89%, 0.76% and 1.1% for 10%, 50% and 90% delay estimations, respectively. In the accuracy comparison of this example, the proposed algorithm can improve the accuracies of 10%, 50% and 90% delay estimations obviously as the Padé order increases, while PRIMA provide less accurate estimations for



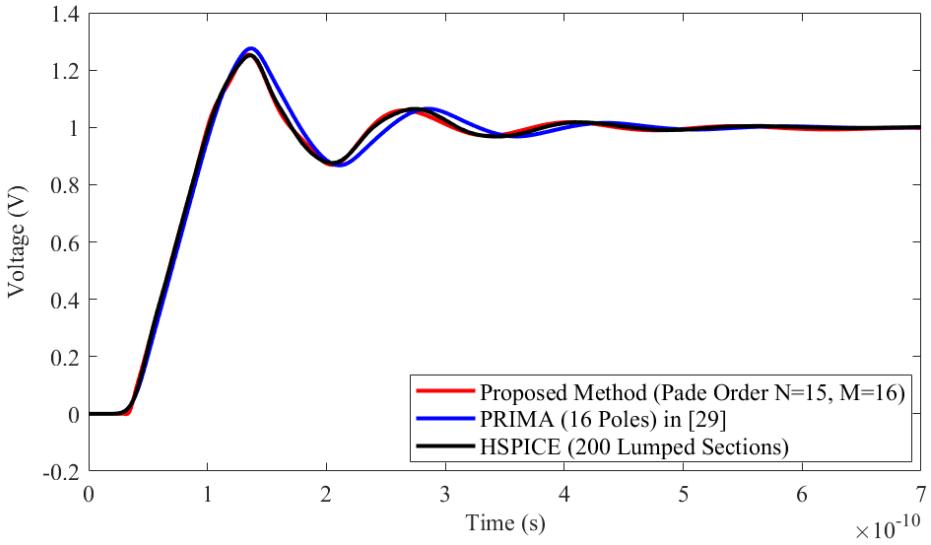
(a)



(b)



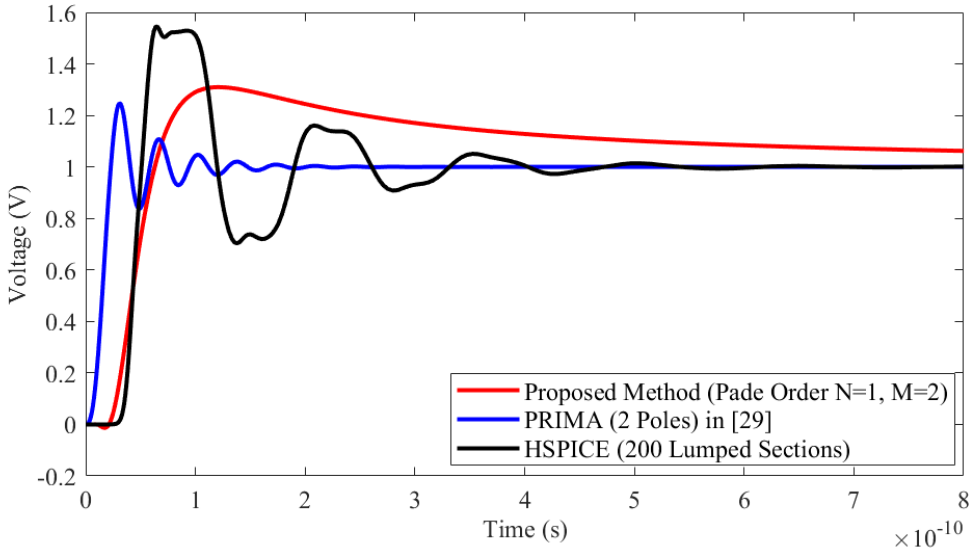
(c)



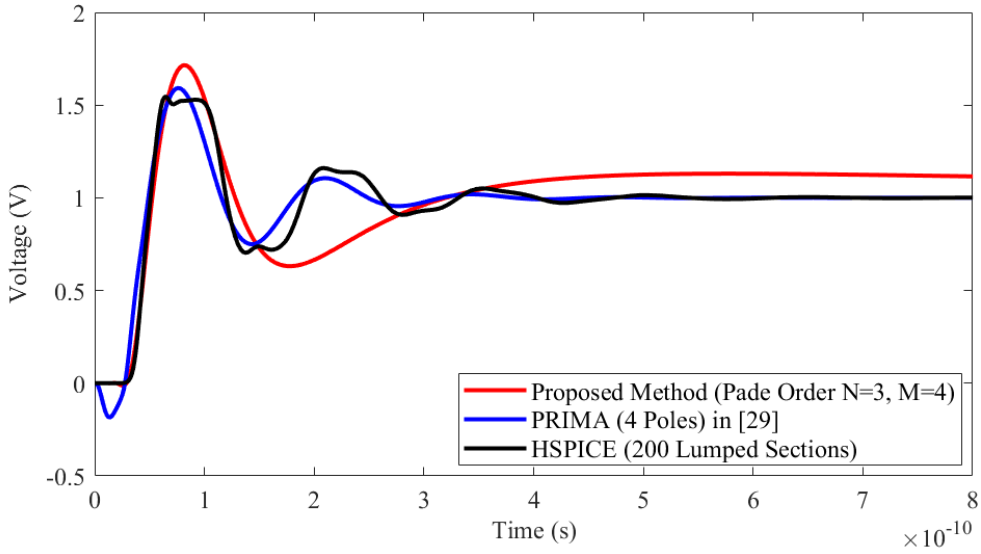
(d)

Figure 4. 16: The far end time domain response at node N_1 for Example 1. Line length = 0.2 cm, line width = 2 µm, per-unit-length parameters (R = 88.29/Ω/cm, L = 15.38 nH/cm, C = 1.8 pF/cm), R_s = 20 Ω, C_l = 10 fF. the input is a ramp signal with rise time of 0.1 ns (a) Comparison of proposed method Padé order (N = 1, M = 2), HSPICE and PRIAM (2 poles) [29] (b) Comparison of proposed method Padé order (N = 3, M = 4), HSPICE and PRIAM (4 poles) [29] (c) Comparison of proposed method Padé order (N = 7, M = 8), HSPICE and PRIAM (8 poles) [29] (d) Comparison of proposed method Padé order (N = 15, M = 16), HSPICE and PRIAM (16 poles) [29]

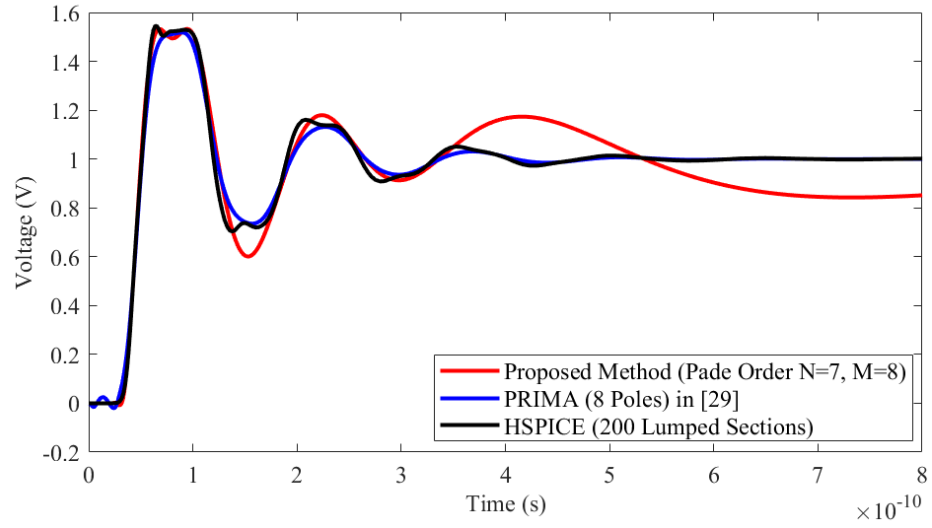
10%, 50% and 90% delay when compared to the proposed algorithm. And the higher approximation order of PRIMA only contributes to improve the accuracy of the steady state not to the accuracies of the 10%, 50% and 90% delay estimations. Observed from the Tables 4.20 and 4.21 it is visible that the overall error of the delay estimations calculated by both the proposed method and PRIMA goes down significantly when the rise time increases, which denotes the input signal becomes ramp.



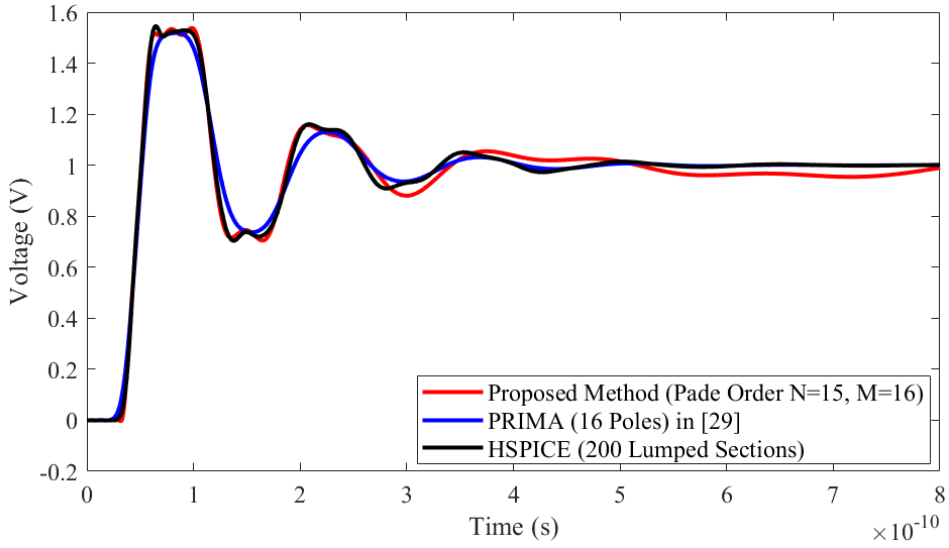
(a)



(b)



(c)



(d)

Figure 4. 17: The far end time domain response at node N_1 for Example 1. Line length = 0.2 cm, line width = 2 µm, per-unit-length parameters (R = 88.29/Ω/cm, L = 15.38 nH/cm, C = 1.8 pF/cm), R_s = 20 Ω, C_l = 10 fF. the input is a ramp signal with rise time of 0.025 ns (a) Comparison of proposed method Padé order (N = 1, M = 2), HSPICE and PRIAM (2 poles) [29] (b) Comparison of proposed method Padé order (N = 3, M = 4), HSPICE and PRIAM (4 poles) [29] (c) Comparison of proposed method Padé order (N = 7, M = 8), HSPICE and PRIAM (8 poles) [29] (d) Comparison of proposed method Padé order (N = 15, M = 16), HSPICE and PRIAM (16 poles) [29]

Table 4. 20: Comparisons of 10%, 50% and 90% delays at node N_1 using the proposed method and PRIMA [29] for Example 1 when the rise time is 0.1 ns and line length is 0.2 cm

Width	R_s	C_l	HSPICE			PRIMA [29] (Approximation Order)						Proposed Method (Padé Order N, M)					
						2			4			(1, 2)			(3, 4)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
(μm)	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
2	20	10	41.25	67.48	93.76	0.14	0.51	0.90	40.64	68.15	94.64	38.70	70.58	114.54	41.62	67.17	92.14
	50	50	46.14	79.76	112.23	22.83	49.93	96.75	42.95	78.93	111.68	44.22	83.74	140.41	46.85	79.21	111.42
	100	100	52.90	98.49	144.17	22.93	63.98	99.15	49.65	98.83	153.35	51.62	104.72	188.28	53.63	97.14	149.35
6	20	10	50.17	77.64	104.85	13.08	50.16	89.10	46.95	76.26	102.84	46.20	80.51	125.76	50.53	77.90	102.34
	50	50	56.07	92.05	128.36	21.99	48.41	96.73	51.45	92.65	124.78	52.08	97.55	165.05	55.82	91.45	128.14
	100	100	62.71	115.06	220.96	28.75	62.08	98.43	56.08	114.03	216.90	60.37	125.83	238.38	63.13	113.27	200.26
10	20	10	58.02	86.42	115.05	12.81	49.56	89.13	53.04	85.32	110.22	52.42	89.76	138.68	57.92	87.19	112.66
	50	50	62.85	103.60	144.29	21.49	47.93	96.13	58.66	103.74	149.75	58.97	111.24	192.71	63.64	102.71	146.92
	100	100	70.27	130.91	283.37	28.08	61.38	98.61	64.23	127.59	290.01	68.36	147.97	295.69	71.69	129.76	295.11
Average Error (%)						63.0	41.7	29.5	7.1	1.1	2.7	5.6	6.6	21.4	1.0	0.85	2.9
Maximum Error (%)						77.9	53.7	65.2	10.6	2.5	6.4	9.7	13.0	33.6	2.0	1.6	9.4

Table 4. 20 (Cont'd.):

Width	R_s	C_l	HSPICE			PRIMA [29] (Approximation Order)						Proposed Method (Padé Order N, M)					
						8			16			(7, 8)			(15, 16)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
(µm)	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
2	20	10	41.25	67.48	93.76	42.39	69.55	95.94	42.39	69.55	95.94	41.22	67.76	93.19	40.79	67.38	93.52
	50	50	46.14	79.76	112.23	46.18	80.30	113.60	46.18	80.30	113.60	46.17	80.05	112.03	46.13	79.76	112.40
	100	100	52.90	98.49	144.17	52.85	98.85	145.98	52.85	98.85	145.98	52.77	98.58	145.48	52.86	98.54	144.35
6	20	10	50.17	77.64	104.85	49.80	77.94	105.15	49.80	77.94	105.15	50.05	77.67	104.88	50.14	77.64	104.91
	50	50	56.07	92.05	128.36	55.18	92.61	128.51	55.18	92.61	128.51	55.59	92.51	127.55	56.02	92.04	128.45
	100	100	62.71	115.06	220.96	62.10	115.90	219.55	62.10	115.90	219.55	61.13	115.13	227.25	62.52	115.09	220.96
10	20	10	58.02	86.42	115.05	57.21	86.91	116.88	57.21	86.91	116.88	58.61	86.41	115.58	57.93	86.55	115.08
	50	50	62.85	103.60	144.29	63.00	103.90	143.80	63.00	103.90	143.80	63.00	104.20	144.03	62.48	103.55	143.70
	100	100	70.27	130.91	283.37	70.35	131.13	289.23	70.35	131.13	289.23	70.15	130.61	284.82	70.23	131.12	283.85
Average Error (%)						0.89	0.76	1.1	0.89	0.76	1.1	0.60	0.25	0.71	0.27	0.066	0.14
Maximum Error (%)						2.8	3.1	2.3	2.8	3.1	2.3	2.5	0.58	2.8	1.1	0.16	0.41

Table 4. 21: Comparisons of 10%, 50% and 90% delays at node N_1 using the proposed method and PRIMA [29] for Example 1 when the rise time is 0.025 ns and line length is 0.2 cm

Width	R_s	C_l	HSPICE			PRIMA [29] (Approximation Order)						Proposed Method (Padé Order N, M)					
						2			4			(1, 2)			(3, 4)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
(μm)	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
2	20	10	36.03	42.74	49.14	7.60	15.08	21.00	30.24	37.28	46.80	27.96	42.16	57.59	34.18	43.13	50.47
	50	50	38.66	48.34	57.10	13.52	25.04	32.60	33.06	41.42	58.72	31.26	50.48	75.47	37.40	48.96	59.76
	100	100	41.84	56.18	97.53	17.50	33.15	45.65	36.90	49.58	98.23	35.53	64.64	114.77	40.99	57.65	80.94
6	20	10	45.40	51.99	58.41	7.38	14.55	20.37	33.24	44.91	55.38	33.96	51.22	70.34	41.99	52.15	60.77
	50	50	47.32	57.36	67.18	13.02	24.06	31.22	35.31	45.89	73.05	37.43	61.76	96.90	45.05	58.43	72.13
	100	100	50.18	65.69	167.88	16.71	31.71	43.47	39.42	55.74	177.54	42.30	81.54	162.65	48.86	68.73	244.22
10	20	10	52.50	59.66	66.37	7.20	14.25	20.16	35.67	49.56	64.08	38.99	59.28	82.56	48.32	59.78	69.83
	50	50	54.56	65.21	76.79	12.74	23.53	30.53	58.66	103.74	149.75	42.87	72.89	121.20	51.45	67.03	84.82
	100	100	57.15	74.56	272.36	16.32	30.99	42.48	40.89	91.80	247.72	48.52	100.13	221.13	55.56	79.83	315.29
Average Error (%)						73.3	59.4	62.3	20.4	20.7	15.1	20.1	11.2	26.2	4.6	2.4	12.5
Maximum Error (%)						86.3	76.1	84.4	32.1	59.1	95.0	25.7	34.3	57.8	8.0	7.1	45.5

Table 4.21 (Cont'd.):

Width	R_s	C_l	HSPICE			PRIMA [29] (Approximation Order)						Proposed Method (Padé Order N, M)					
						8			16			(7, 8)			(15, 16)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
(μm)	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
2	20	10	36.03	42.74	49.14	33.64	43.00	49.40	33.64	43.00	49.40	35.68	42.81	48.76	35.93	42.53	49.26
	50	50	38.66	48.34	57.10	37.86	48.30	57.68	37.86	48.30	57.68	38.62	48.26	56.78	38.57	48.36	56.84
	100	100	41.84	56.18	97.53	41.10	56.15	98.98	41.10	56.15	98.98	41.84	55.84	100.94	41.63	55.99	97.15
6	20	10	45.40	51.99	58.41	43.26	51.48	58.56	43.26	51.48	58.56	44.38	52.11	58.10	44.72	51.74	58.12
	50	50	47.32	57.36	67.18	46.13	56.96	68.57	46.13	56.96	68.57	47.24	57.27	66.74	47.32	57.35	66.62
	100	100	50.18	65.69	167.88	48.81	66.06	167.52	48.81	66.06	167.52	50.57	65.28	163.32	50.31	65.40	165.34
10	20	10	52.50	59.66	66.37	49.89	58.50	66.69	49.89	58.50	66.69	51.57	59.10	66.16	52.14	59.79	66.70
	50	50	54.56	65.21	76.79	52.93	64.74	80.06	52.93	64.74	80.06	54.41	65.16	76.61	54.58	65.21	76.07
	100	100	57.15	74.56	272.36	55.41	75.90	265.08	55.41	75.90	265.08	57.87	74.22	304.08	57.61	74.12	271.50
Average Error (%)						3.5	0.83	1.4	3.5	0.83	1.4	0.84	0.38	2.3	0.48	0.29	0.63
Maximum Error (%)						6.6	1.9	4.3	6.6	1.9	4.3	2.2	0.94	11.6	1.5	0.59	1.5

To verify the efficiency of the proposed method, the average run times to solve for node N_1 at 500 time points using both the proposed algorithm and PRIMA are calculated and presented as Table 4.22. It is shown from Table 4.22 that the proposed algorithm needs less run time than PRIMA for rise time of 0.1 ns and 0.025 ns in this example. The proposed algorithm is 20 to 21 times faster than PRIMA for approximation order of 16.

Table 4. 22: Run Time Comparison between the Proposed Method and PRIMA [29] for Example 1

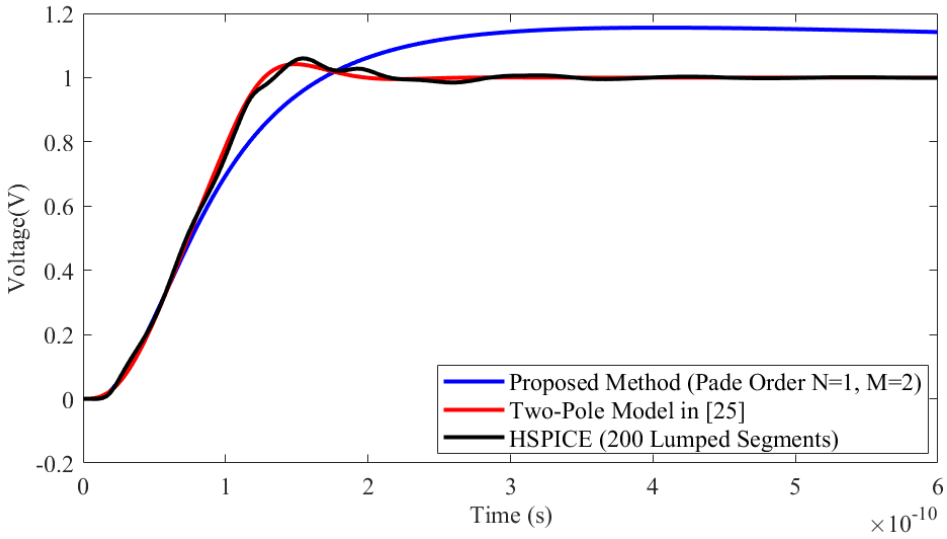
Rise Time (ns)	PRIMA [29]		Proposed Method		Speed Up
	Approximation Order	Run Time (ms)	Padé Order	Run Time (ms)	
0.1	2	247.13	(1, 2)	4.09	60
	4	255.69	(3, 4)	5.52	46
	8	257.12	(7, 8)	7.57	34
	16	265.88	(15, 16)	12.63	21
0.025	2	299.52	(1, 2)	3.69	81
	4	262.43	(3, 4)	5.76	46
	8	266.78	(7, 8)	8.43	32
	16	270.76	(15, 16)	13.25	20

4.2.2 Example 2- Symmetrical Unbalanced Distributed *RLC* Tree

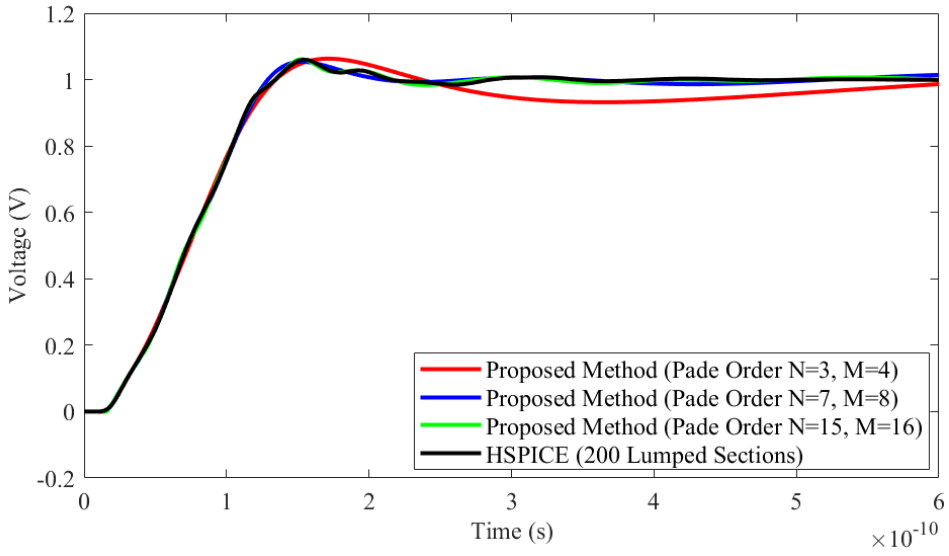
4.2.2.1 Comparing the Proposed Algorithm to Two-Pole Model

In this example, the symmetrical unbalanced distributed interconnect structure discussed in Section 4.1.2 is analyzed for rise ramp input with rise time of 0.1 ns and 0.025 ns, respectively. The 10%, 50% and 90% delays at node N_7 (Figure 4.7) are calculated using NILT, HSPICE, the two-pole model [25] and PRIMA [29] for various line lengths, resistive loads and capacitive loads. Tables 4.23 and 4.24 show the results of a unit rising ramp input with 0.1 ns and 0.025 ns rise time when using the two-pole model and the proposed algorithm, respectively. Figure 4.18 (a) and Figure 4.19 (a) show the transient responses comparing NILT, Padé order (N = 1, M = 2), HSPICE and the two-pole model for x = 2, R_s = 10 Ω, and C_x = 20 fF when the input voltage is a ramp with rise time of 0.1 ns and

0.025ns, respectively. Figure 4.18 (b) and Figure 4.19 (b) show the NILT responses of the higher order Padé approximations.

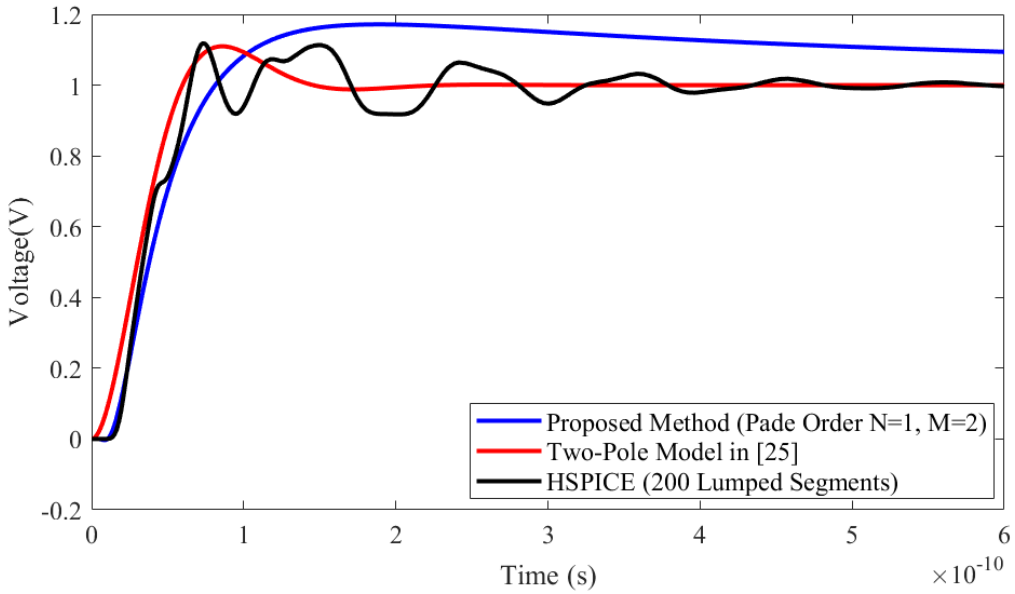


(a)

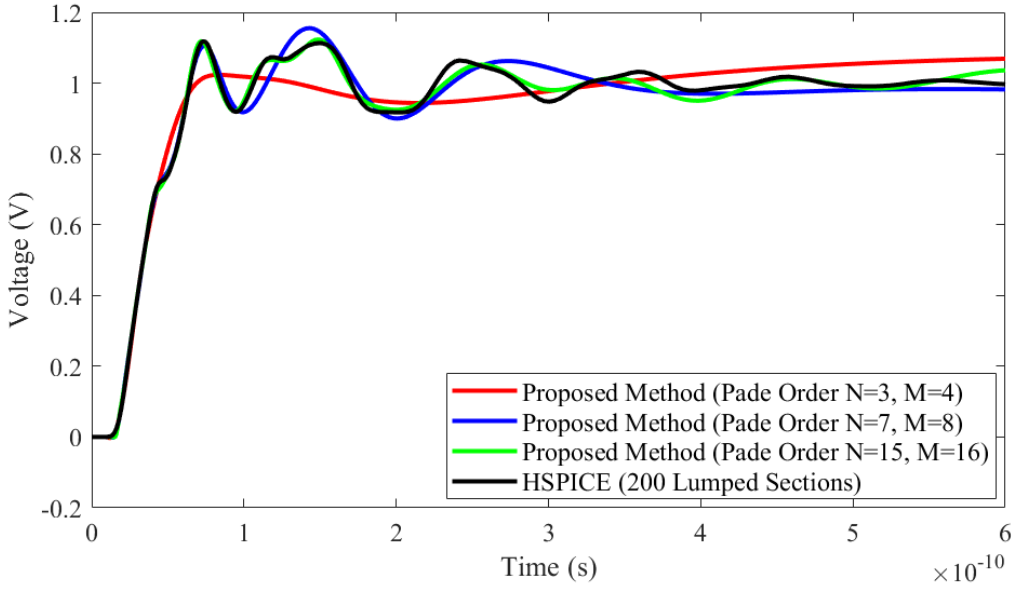


(b)

Figure 4. 18: The far end time domain response at node N_7 for Example 2 with unit rising ramp input of 0.1 ns rise time, x = 2, R_s = 10 Ω, C_x= 20 fF (a) Comparison of proposed method Padé order (N = 1, M = 2), HSPICE and two-pole model [25] (b) Comparison of proposed method Padé orders (N = 3, M = 4), (N = 7, M = 8) and (N = 15, M = 16), respectively, against HSPICE



(a)



(b)

Figure 4. 19: The far end time domain response at node N_7 for Example 2 with unit rising ramp input of 0.025 ns rise time, x = 2, R_s = 10 Ω, C_x= 20 fF (a) Comparison of proposed method Padé order (N = 1, M = 2), HSPICE and two-pole model [25] (b) Comparison of proposed method Padé orders (N = 3, M = 4), (N = 7, M = 8) and (N = 15, M = 16), respectively, against HSPICE

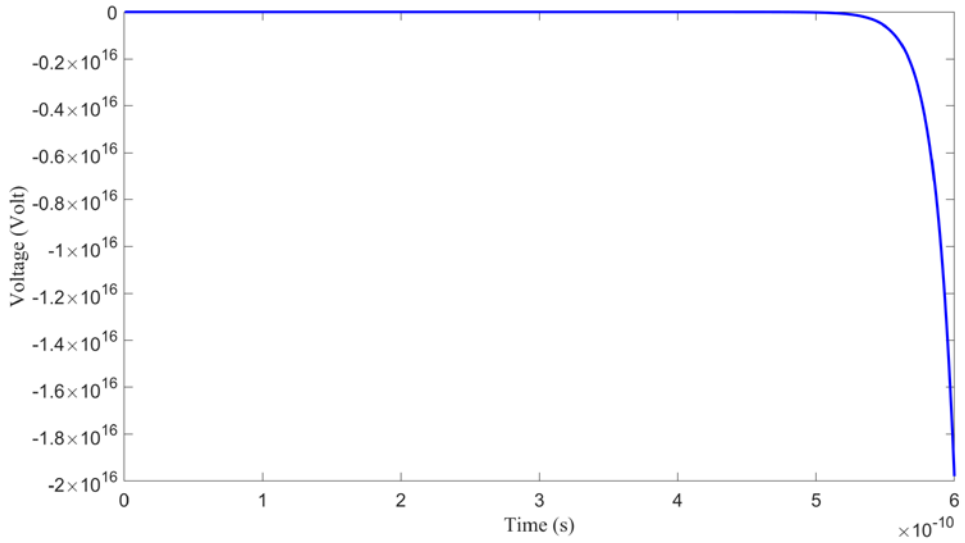


Figure 4. 20: The far end time domain response at N_7 for a unit ramp input with rise time of 0.1 ns using an unstable three-pole model, x=2, R_s = 10 Ω, C_x= 20 fF

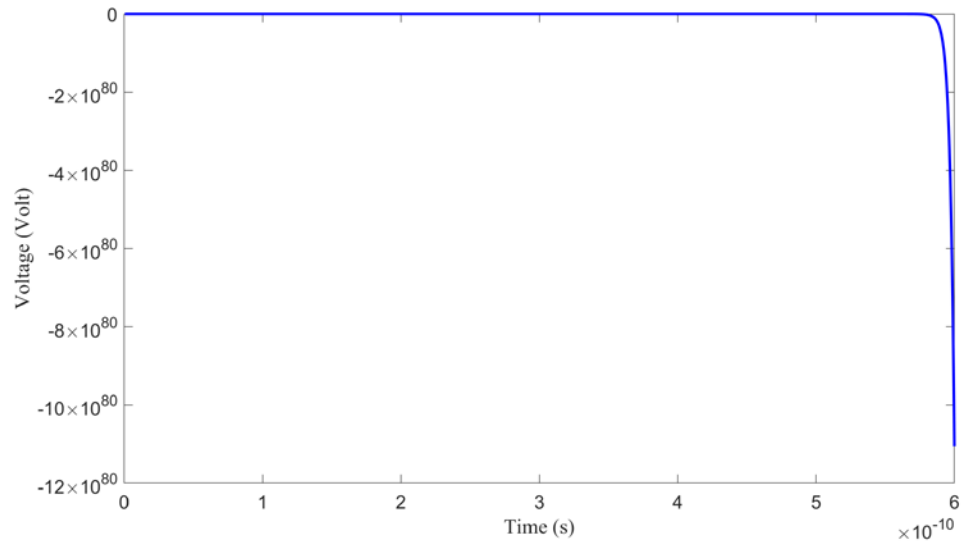


Figure 4. 21: The far end time domain response at N_7 for a unit ramp input with rise time of 0.025 ns using an unstable three-pole model, x=2, R_s = 10 Ω, C_x= 20 fF

Comparing Tables 4.23 and 4.24 demonstrates that as the rise time decreases both NILT and the two-pole model become less accurate. For the rising ramp input, the two-pole model provides better estimates for the 50% and 90% delays than NILT Padé order (N = 1, M = 2), while NILT Padé order (N = 1, M = 2) provides better estimates for the

Table 4. 23: Comparison of 10%, 50% and 90% delays at node N_7 using the proposed algorithm and two-pole model [25] for Example 2 with unit rising ramp input of 0.1 ns rise time

x	R_s	C_x	HSPICE			Two-Pole Model [25]			Proposed Method (Padé Order N, M)											
									(1, 2)			(3, 4)			(7, 8)			(15, 16)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
2	10	20	30.39	72.84	113.28	32.60	73.80	112.20	31.79	75.89	120.09	30.37	73.73	116.26	30.18	72.42	114.70	30.06	72.15	113.36
	10	100	31.44	79.04	123.24	34.60	79.20	118.80	33.53	81.40	140.48	31.46	79.48	125.19	31.30	78.09	125.36	31.19	77.91	123.81
	30	20	39.13	100.39	183.11	41.00	99.60	183.60	40.17	104.87	206.22	38.89	100.42	178.51	39.87	100.99	183.57	39.58	100.46	182.33
	30	100	40.19	109.01	207.95	43.60	108.00	214.80	42.89	115.71	233.32	40.32	109.85	208.54	40.22	109.67	208.45	40.17	108.48	206.33
4	10	20	30.36	84.16	152.05	34.80	84.60	130.80	32.26	85.43	165.02	30.30	83.56	144.65	30.19	83.97	154.06	30.07	83.62	150.33
	10	100	31.21	89.65	164.55	35.60	88.80	139.80	33.16	89.33	175.78	31.52	87.84	156.51	31.29	89.56	168.89	31.19	89.17	162.95
	30	20	39.72	115.27	257.14	42.00	119.40	264.60	41.92	127.92	276.03	38.70	120.05	267.94	39.50	119.79	252.60	39.68	115.56	257.22
	30	100	40.10	122.24	283.92	44.40	127.80	297.60	43.08	137.26	304.48	39.40	129.59	301.88	39.89	124.69	284.81	39.93	126.97	285.87
Average Error (%)						9.5	1.6	5.6	5.8	5.4	9.4	0.82	1.9	3.4	0.62	1.2	1.2	0.58	1.0	0.57
Maximum Error (%)						14.6	4.5	15.0	7.4	12.3	14.0	2.6	6.0	6.3	1.9	3.9	2.6	1.2	3.9	1.1

Table 4. 24: Comparison of 10%, 50% and 90% delays at node N_7 using the proposed algorithm and two-pole model [25] for Example 2 with unit rising ramp input of 0.025 ns rise time

x	R_s	C_x	HSPICE			Two-Pole Model [25]			Proposed Method (Padé order N, M)											
									(1, 2)			(3, 4)			(7, 8)			(15, 16)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
2	10	20	20.18	34.81	61.56	18.88	37.80	56.40	18.57	37.86	67.35	19.18	33.66	56.23	19.89	33.73	59.94	19.77	33.78	60.04
	10	100	21.48	35.95	72.33	18.92	42.00	65.40	18.70	41.34	87.93	21.43	34.62	74.20	21.16	34.49	71.95	21.12	34.52	71.79
	30	20	22.05	60.14	130.91	21.86	55.80	138.60	21.56	56.35	146.13	22.01	53.03	144.61	21.45	58.34	125.35	21.55	58.60	131.21
	30	100	23.30	70.48	161.12	24.98	65.40	171.60	22.52	67.56	184.55	23.32	69.52	172.53	22.84	69.95	148.67	22.89	69.96	160.41
4	10	20	20.19	33.74	85.69	22.96	44.40	77.40	18.62	40.91	92.43	19.01	33.40	87.92	20.02	33.75	85.68	20.17	33.77	85.74
	10	100	21.60	34.53	97.26	23.82	41.04	88.80	19.55	45.54	92.62	21.22	34.50	108.45	21.16	34.55	97.02	21.38	34.51	96.50
	30	20	22.01	82.54	203.45	25.89	77.40	222.60	21.35	74.79	214.44	22.02	86.96	220.74	21.45	81.44	194.17	21.55	82.20	205.04
	30	100	23.00	93.99	238.01	25.96	87.00	256.80	22.32	83.27	245.93	23.33	103.46	247.15	22.84	93.00	219.71	22.89	93.42	238.86
Average Error (%)						10.1	13.0	8.3	6.2	13.5	9.8	1.8	4.6	6.9	1.7	1.6	3.5	1.3	1.4	0.7
Maximum Error (%)						17.6	31.6	9.7	12.9	31.9	21.6	5.8	11.8	11.5	2.7	4.1	7.7	2.3	4.0	2.5

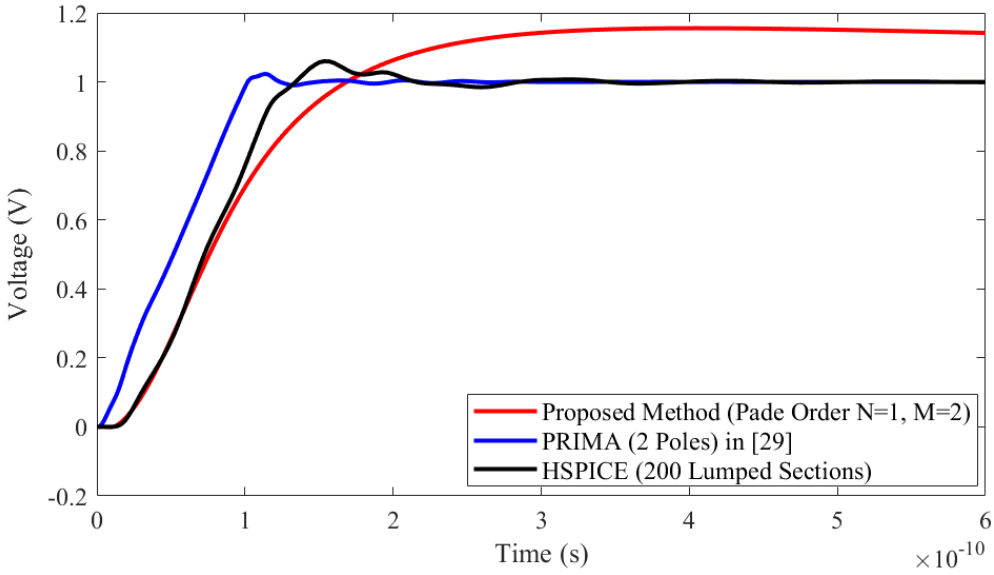
10% delay (Tables 4.23 and 4.24). Typically, as the rise time of the input signal decreases and approaches the unit step response, NILT Padé order (N = 1, M = 2) provides better accuracy than the two-poles model and as the rise time increases the two-pole model becomes more accurate compared to the Padé order (N = 1, M = 2) approximation. Tables 4.23 and 4.24 also provide the 10%, 50% and 90% delay estimates of higher order Padé approximations, illustrating that Padé order (N = 3, M = 4) and higher provides better accuracy than the two-pole model. Conversely, increasing the number of poles by using a higher order Maclaurin series to approximate the cosh and sinh terms of (2.14) following the steps of [25] results in unstable three-pole models for this example shown as Figures 4.20 and 4.21.

The average run times to solve nodes N_1 to N_7 at 500 time points are calculated and compared using the two-pole model and the proposed method for the case when x = 2, respectively. For this example, the average run time for NILT, Padé order (N = 1, M = 2) is 18.94 ms and the average run time for the two-pole model is 5.03 ms when the rise time is 0.1 ns. For 0.025 ns rise time, the average run time for NILT, Padé order (N = 1, M = 2) is 10.47 ms and the average run time for the two-pole model is 5.16 ms. The results show that the two-pole model is 2 to 4 times faster when compared to the proposed algorithm.

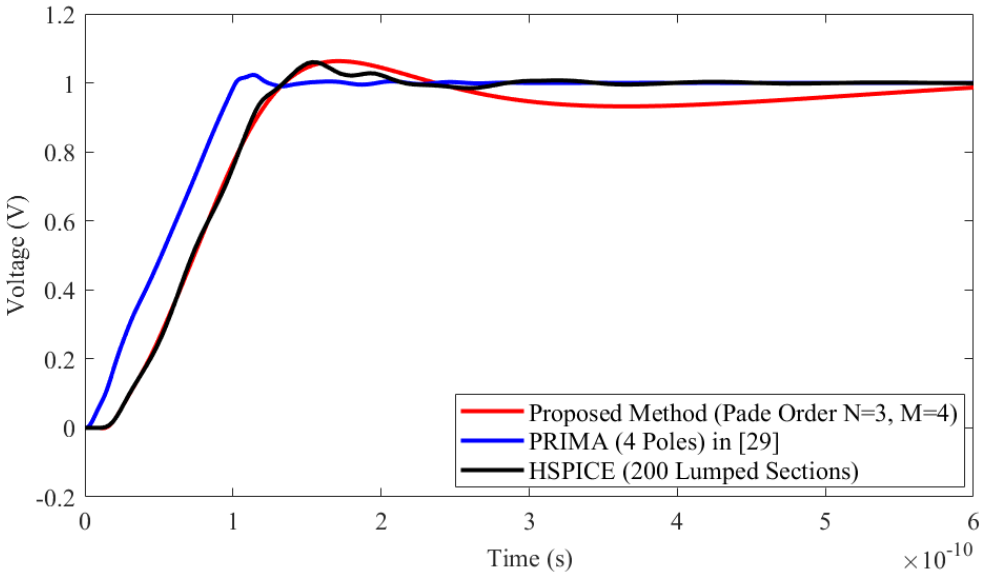
4.2.2.2 Comparing the Proposed Algorithm to PRIMA

Selecting the approximation orders of PRIMA [29] 2, 4, 8 and 16 to correspond to the same Padé orders for the proposed algorithm. As discussed in Section 4.1.2.2, 1, 2 and 4 moments are matched for four different approximation orders of PRIMA for the a five-port network in this example. Figure 4.21 (a) and Figure 4.22 (a) show the transient responses obtained from the NILT with Padé order (N = 1, M = 2), HSPICE and the PRIMA with approximation order 2 for x = 2, R_s = 10 Ω, C_x= 20 fF when the rise times of ramp input are 0.1 ns and 0.025 ns, respectively. Figure 4.21 (b)-(d) and Figure 4.22 (b)-(d) show the comparison of responses of the higher order approximations between the NILT (Padé orders (N = 3, M = 4), (N = 7, M = 8) and (N = 15, M = 16), respectively) and the PRIMA (approximation orders 4, 8 and 16, respectively) for the rise times of ramp input are 0.1 ns and 0.025 ns. From the Figure 4.21 and Figure 4.22, the proposed algorithm with Padé order (N = 1, M = 2) and (N = 3, M = 4) can provide more accurate estimations for 10%,

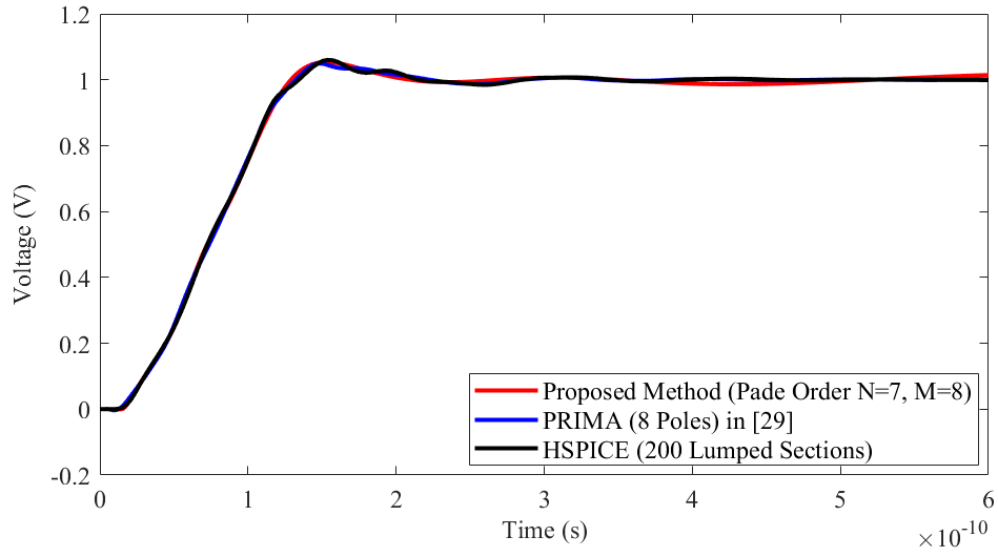
50% and 90% delay compared to PRIMA for approximation orders 2 and 4 for both 0.1 ns and 0.025 ns rise times. Once again, the proposed algorithm can improve the accuracy of delay estimation by increasing the Padé order. The delay results at the node N_7 are calculated using both the proposed algorithm and PRIMA and compared to those of HSPICE for rise time of 0.1 ns and 0.025 ns, respectively (Tables 4.25 and 4.26).



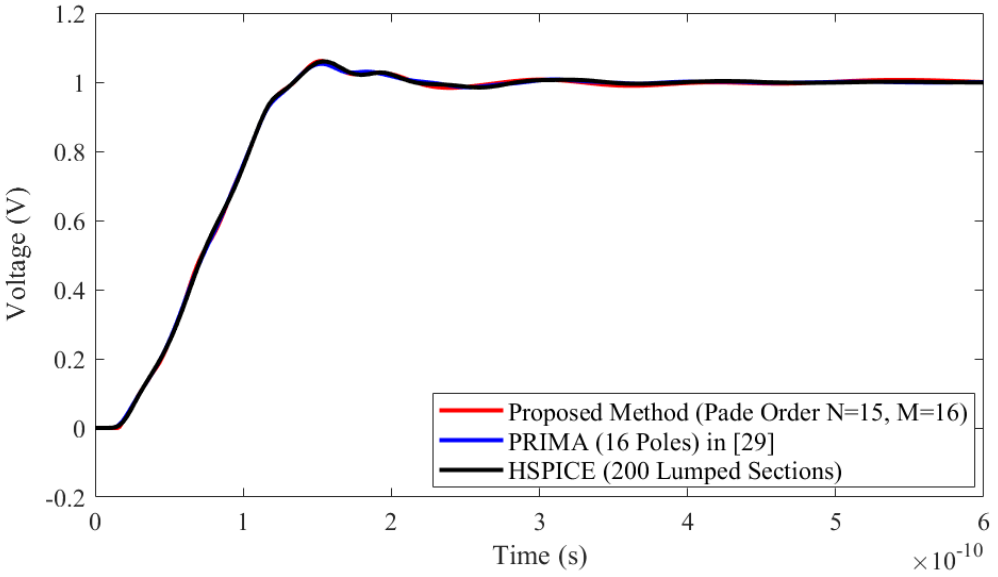
(a)



(b)



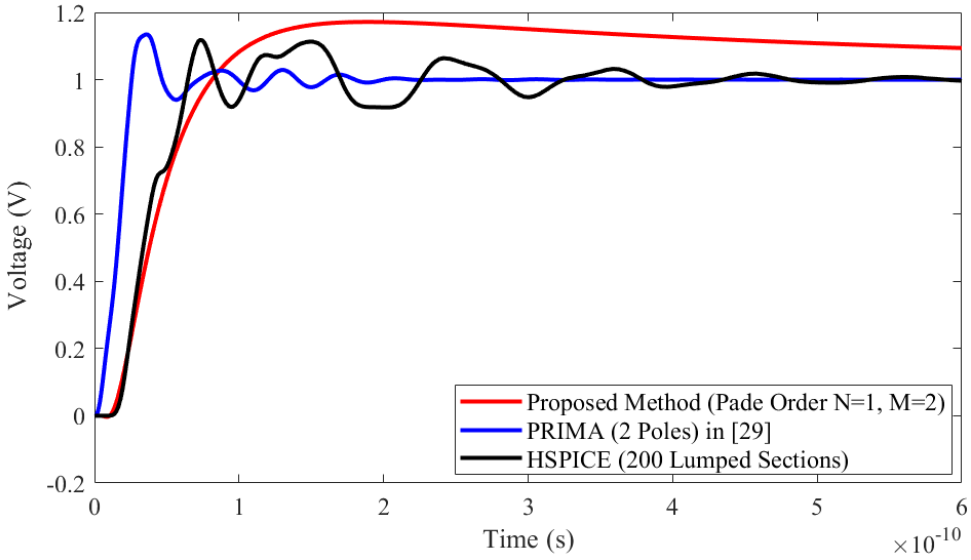
(c)



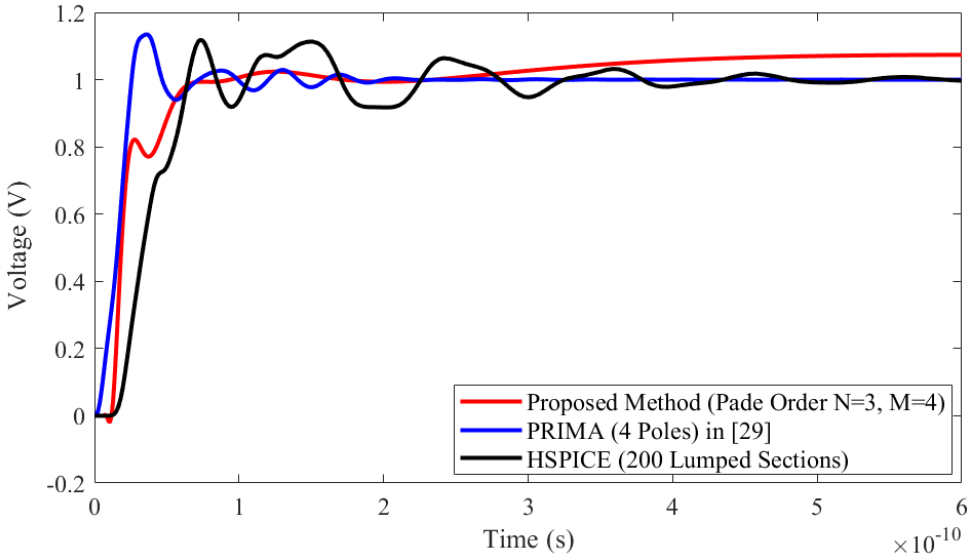
(d)

Figure 4. 22: The far end time domain response at node N_7 for Example 2 with unit rising ramp input of 0.1 ns rise time, x = 2, R_s = 10 Ω, C_x= 20 fF (a) Comparison of proposed method Padé order (N = 1, M = 2), HSPICE and PRIAM (2 poles) [29] (b) Comparison of proposed method Padé order (N = 3, M = 4), HSPICE and PRIAM (4 poles) [29] (c) Comparison of proposed method Padé order (N = 7, M = 8), HSPICE and PRIAM (8 poles) [29] (d) Comparison of proposed method Padé order (N = 15, M = 16), HSPICE and PRIAM (16 poles) [29]

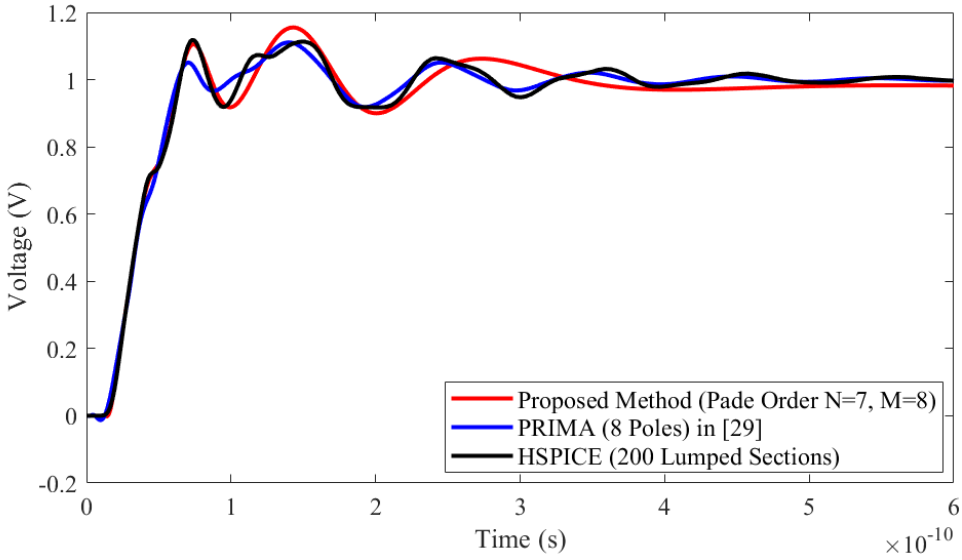
Observed from Tables 4.25 and 4.26, the proposed method with Padé order (N = 1, M = 2) provides more accurate delay estimations compared to PRIMA for approximation order 2. Since approximation orders of 2 and 4 for PRIMA match only one moment, the accuracy of the delay estimations is not improved though the approximation order of PRIMA is increased from 2 to 4. Conversely, the accuracies of 10%, 50% and 90% delay



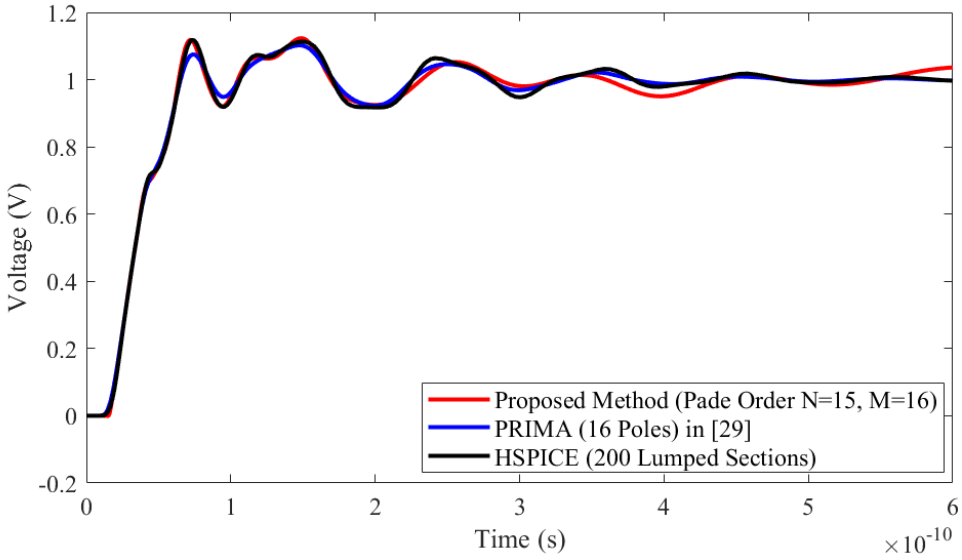
(a)



(b)



(c)



(d)

Figure 4. 23: The far end time domain response at node N_7 for Example 2 with unit rising ramp input of 0.025 ns rise time, x = 2, R_s = 10 Ω, C_x= 20 fF (a) Comparison of proposed method Padé order (N = 1, M = 2), HSPICE and PRIAM (2 poles) [29] (b) Comparison of proposed method Padé order (N = 3, M = 4), HSPICE and PRIAM (4 poles) [29] (c) Comparison of proposed method Padé order (N = 7, M = 8), HSPICE and PRIAM (8 poles) [29] (d) Comparison of proposed method Padé order (N = 15, M = 16), HSPICE and PRIAM (16 poles) [29]

Table 4. 25: Comparisons of 10%, 50% and 90% delays at node N_7 using the proposed method and PRIMA [29] for Example 2 when the rise time is 0.1 ns and line length is 0.2 cm

Width	R_s	C_x	HSPICE			PRIMA [29] (Approximation Order)						Proposed Method (Padé order N, M)					
						2			4			(1, 2)			(3, 4)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
2	10	10	30.39	72.84	113.28	13.92	51.63	90.96	13.92	51.63	90.96	31.79	75.89	120.09	30.37	73.73	116.26
	10	10	31.44	79.04	123.24	18.36	57.33	96.03	18.36	57.33	96.03	33.53	81.40	140.48	31.46	79.48	125.19
	30	30	39.13	100.39	183.11	15.54	53.22	93.27	15.54	53.22	93.27	40.17	104.87	206.22	38.89	100.42	178.51
	30	30	40.19	109.01	207.95	23.25	68.82	109.95	23.25	68.82	109.95	42.89	115.71	233.32	40.32	109.85	208.54
4	10	10	30.36	84.16	152.05	13.32	51.30	91.83	13.32	51.30	91.83	32.26	85.43	165.02	30.30	83.56	144.65
	10	10	31.21	89.65	164.55	15.84	58.65	96.51	15.84	58.65	96.51	33.16	89.33	175.78	31.52	87.84	156.51
	30	30	39.72	115.27	257.14	15.18	53.43	94.14	15.18	53.43	94.14	41.92	127.92	276.03	38.70	120.05	267.94
	30	30	40.10	122.24	283.92	18.03	69.78	119.31	18.03	69.78	119.31	43.08	137.26	304.48	39.40	129.59	301.88
Average Error (%)						52.6	38.8	42.5	52.6	38.8	42.5	5.8	5.4	9.4	0.82	1.9	3.4
Maximum Error (%)						61.8	53.6	63.4	61.8	53.6	63.4	7.4	12.3	14.0	2.6	6.0	6.3

Table 4.25 (Cont'd.):

Width	R_s	C_x	HSPICE			PRIMA [29] (Approximation Order)						Proposed Method (Padé order N, M)					
						8			16			(7, 8)			(15, 16)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
2	10	10	30.39	72.84	113.28	30.63	73.71	113.55	30.63	73.71	113.55	30.18	72.42	114.70	30.06	72.15	113.36
	10	10	31.44	79.04	123.24	31.08	78.21	123.93	31.08	78.21	123.93	31.30	78.09	125.36	31.19	77.91	123.81
	30	30	39.13	100.39	183.11	40.20	100.35	182.34	40.20	100.35	182.34	39.87	100.99	183.57	39.58	100.46	182.33
	30	30	40.19	109.01	207.95	41.76	108.39	209.34	41.76	108.39	209.34	40.22	109.67	208.45	40.17	108.48	206.33
4	10	10	30.36	84.16	152.05	28.53	81.75	148.71	28.53	81.75	148.71	30.19	83.97	154.06	30.07	83.62	150.33
	10	10	31.21	89.65	164.55	29.97	86.37	156.84	29.97	86.37	156.84	31.29	89.56	168.89	31.19	89.17	162.95
	30	30	39.72	115.27	257.14	40.74	115.62	251.82	40.74	115.62	251.82	39.50	119.79	252.60	39.68	115.56	257.22
	30	30	40.10	122.24	283.92	39.03	131.34	281.85	39.03	131.34	281.85	39.89	124.69	284.81	39.93	126.97	285.87
Average Error (%)						3.0	2.1	1.4	3.0	2.1	1.4	0.62	1.2	1.2	0.58	1.0	0.57
Maximum Error (%)						6.0	7.4	4.7	6.0	7.4	4.7	1.9	3.9	2.6	1.2	3.9	1.1

Table 4. 26: Comparisons of 10%, 50% and 90% delays at node N_7 using the proposed method and PRIMA [29] for Example 2 when the rise time is 0.025 ns and line length is 0.2 cm

Width	R_s	C_x	HSPICE			PRIMA [29] (Approximation Order)						Proposed Method (Padé order N, M)					
						2			4			(1, 2)			(3, 4)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
2	10	10	20.18	34.81	61.56	5.61	16.02	23.52	5.61	16.02	23.52	18.57	37.86	67.35	19.18	33.66	56.23
	10	10	21.48	35.95	72.33	9.03	21.78	40.23	9.03	21.78	40.23	18.70	41.34	87.93	21.43	34.62	74.20
	30	30	22.05	60.14	130.91	5.91	17.73	26.61	5.91	17.73	26.61	21.56	56.35	146.13	22.01	53.03	144.61
	30	30	23.30	70.48	161.12	9.63	29.22	54.33	9.63	29.22	54.33	22.52	67.56	184.55	23.32	69.52	172.53
4	10	10	20.19	33.74	85.69	17.49	32.52	74.52	17.49	32.52	74.52	18.62	40.91	92.43	19.01	33.40	87.92
	10	10	21.60	34.53	97.26	8.28	18.30	48.03	8.28	18.30	48.03	19.55	45.54	92.62	21.22	34.50	108.45
	30	30	22.01	82.54	203.45	5.25	17.88	27.69	5.25	17.88	27.69	21.35	74.79	214.44	22.02	86.96	220.74
	30	30	23.00	93.99	238.01	8.70	22.14	60.66	8.70	22.14	60.66	22.32	83.27	245.93	23.33	103.46	247.15
Average Error (%)						59.4	53.5	59.6	59.4	53.5	59.6	6.2	13.5	9.8	1.8	4.6	6.9
Maximum Error (%)						76.1	78.3	86.4	76.1	78.3	86.4	12.9	31.9	21.6	5.8	11.8	11.5

Table 4.26 (Cont'd.):

Width	R_s	C_x	HSPICE			PRIMA [29] (Approximation Order)						Proposed Method (Padé order N, M)					
						8			16			(7, 8)			(15, 16)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
2	10	10	20.18	34.81	61.56	18.39	33.96	57.09	18.39	33.96	57.09	19.89	33.73	59.94	19.77	33.78	60.04
	10	10	21.48	35.95	72.33	19.83	34.50	69.33	19.83	34.50	69.33	21.16	34.49	71.95	21.12	34.52	71.79
	30	30	22.05	60.14	130.91	20.31	54.93	127.59	20.31	54.93	127.59	21.45	58.34	125.35	21.55	58.60	131.21
	30	30	23.30	70.48	161.12	21.78	67.05	157.53	21.78	67.05	157.53	22.84	69.95	148.67	22.89	69.96	160.41
4	10	10	20.19	33.74	85.69	18.81	33.93	85.74	18.81	33.93	85.74	20.02	33.75	85.68	20.17	33.77	85.74
	10	10	21.60	34.53	97.26	18.75	33.33	87.36	18.75	33.33	87.36	21.16	34.55	97.02	21.38	34.51	96.50
	30	30	22.01	82.54	203.45	18.99	70.74	207.21	18.99	70.74	207.21	21.45	81.44	194.17	21.55	82.20	205.04
	30	30	23.00	93.99	238.01	20.40	83.43	232.47	20.40	83.43	232.47	22.84	93.00	219.71	22.89	93.42	238.86
Average Error (%)						9.5	6.2	3.8	9.5	6.2	3.8	1.7	1.6	3.5	1.3	1.4	0.7
Maximum Error (%)						13.7	14.3	10.2	13.7	14.3	10.2	2.7	4.1	7.7	2.3	4.0	2.5

Table 4. 27: Run time comparison between the proposed method and PRIMA [29] for Example 2

Rise Time (ns)	PRIMA [29]		Proposed Method		Speed Up
	Approximation Order	Run Time (ms)	Padé Order	Run Time (ms)	
0.1	2	387.34	(1, 2)	18.94	20
	4	387.34	(3, 4)	29.23	13
	8	408.03	(7, 8)	49.29	8
	16	450.18	(15, 16)	93.87	5
0.025	2	595.14	(1, 2)	10.47	57
	4	595.14	(3, 4)	15.45	39
	8	625.54	(7, 8)	26.52	24
	16	653.78	(15, 16)	50.39	13

estimations are improved significantly when the Padé order is increased from (N = 1, M = 2) to (N = 3, M = 4) (Tables 4.25 and 4.26). For PRIMA, when the approximation order is increased to 8, the accuracies of delay estimations are improved significantly compared to the lower approximation orders, which dropped to 3.0%, 2.1% and 1.4% for 0.1 ns rise time, and 9.5%, 6.2% and 3.8% for 0.025 ns rise time, respectively. Further increasing the approximation order to 16, the accuracies of the delay estimations keep unchanged comparing with approximation order of 8 though the accuracy of steady state is improved. As a comparison, the proposed algorithm improves the accuracies of delay estimations obviously when the Padé order increases. For Padé order (N = 15, M = 16), the average errors of 10%, 50% and 90% delay are 0.58%, 1.0% and 0.57% when the rise time is 0.1 ns, and 1.3%, 1.4% and 0.7% when the rise time is 0.025 ns, respectively.

For efficiency comparison between the proposed algorithm and PRIMA [29], the average run times used to calculate the transient responses for nodes N_1 to N_7 at 500 time points are listed in Table 4.27 for the case when x = 2. As seen from Table 4.27, the proposed method requires less run time than PRIMA when performing the comparison for the same approximation order. The proposed algorithm with Padé order (N = 15, M = 16)

is about 5-13 times faster than PRIMA with approximation order of 16 when the rise times are 0.1 ns and 0.025 ns, respectively. In this example, the proposed algorithm shows advantage over PRIMA in both accuracy of delay estimations and efficiency for complicated interconnect tree structure. The proposed algorithm presents visible superiority in improving the accuracy of delay estimation by improving the Padé orders.

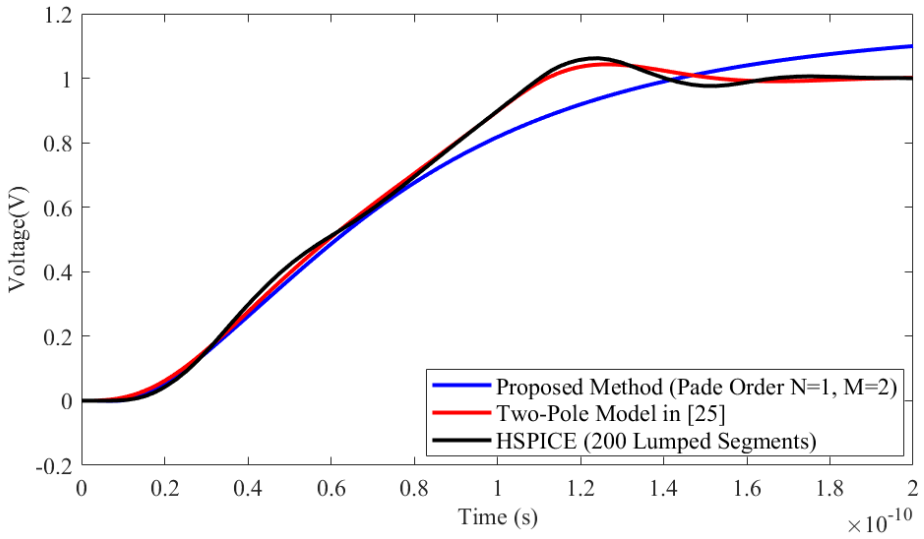
4.2.3 Example 3 - Unsymmetrical Distributed RLC Tree

4.2.3.1 Comparing the Proposed Algorithm to the Two-Pole Model

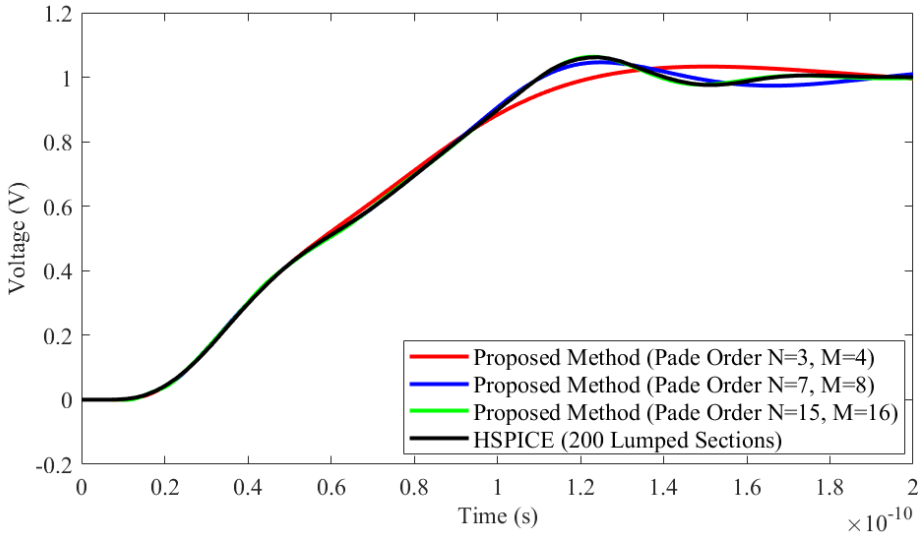
The same unsymmetrical distributed RLC tree structure discussed in Section 4.1.3 is analyzed here with the unit rising ramp input for rise time of 0.1 ns and 0.025 ns, respectively. The 10%, 50% and 90% delays at node N_7 (Figure 4.11) are calculated using NILT, HSPICE and the two-pole model [25] for various line lengths, resistive loads and capacitive loads. Tables 4.28 and 4.29 show the results of a unit rising ramp input with 0.1 ns and 0.025 ns rise time, respectively. Figure 4.24 (a) and Figure 4.25 (a) show the transient responses comparing NILT, Padé order (N = 1, M = 2), HSPICE and the two-pole model for l_x= 0.2 mm, R_s = 10 Ω, and C_x = 20 fF when the input voltage is a unit rising ramp with rise time of 0.1 ns and 0.025ns, respectively. Figure 4.24 (b) and Figure 4.25 (b) show the NILT responses of the higher order Padé approximations. Once again, Tables 4.28 and 4.29 show that as the rise time decreases both NILT and the two-pole model become less accurate. For the unit step response analyzed in Section 4.1.3, NILT Padé order (N = 1, M = 2) is more accurate than the two-pole model (Tables 4.28 and 4.29) but for the rising ramp inputs, the two-pole model provides better estimates for the 50% and 90% delays, while NILT Padé order (N = 1, M = 2) provides better estimates for the 10% delay (Tables 4.28 and 4.29). Also increasing the Padé order of the NILT approximation provides better delay estimates when compared to the two-pole model.

The average run times to solve nodes N_1 to N_9 at 500 time points using NILT and the two-pole model are calculated and compared for the case when l_x = 0.2 mm. For this example, the average run times for the two-pole model are 5.76 ms and 5.83 ms when the rise times are 0.1 ns and 0.025 ns, respectively. For the proposed algorithm with Padé order (N = 1, M = 2), the average run times for are 19.81 ms and 12.98 ms when the rise times

are 0.1 ns and 0.025 ns, respectively. Shown as the results of average run time, the two-pole model is about 2 to 3 times faster when compared to the proposed algorithm for Padé order (N = 1, M = 2).

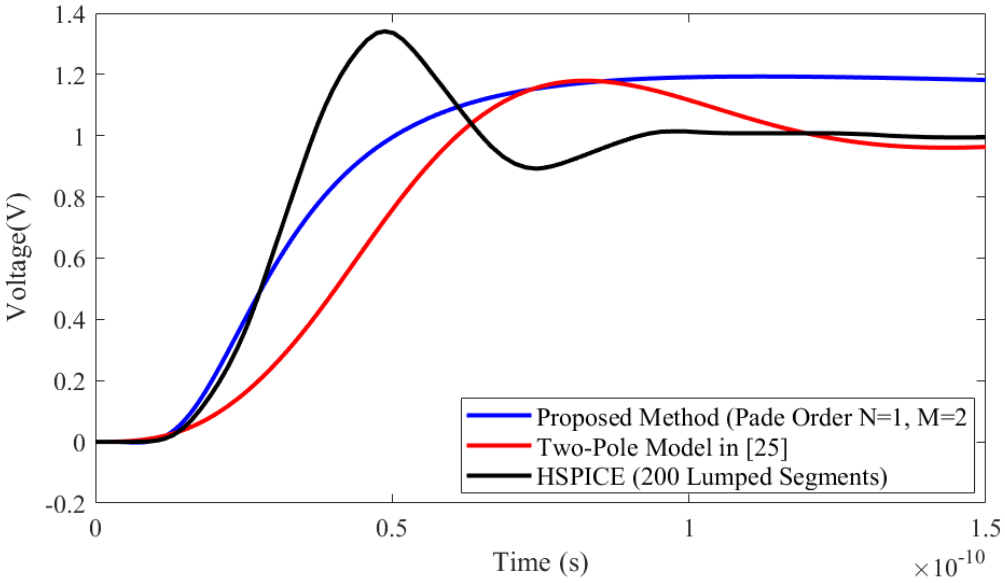


(a)

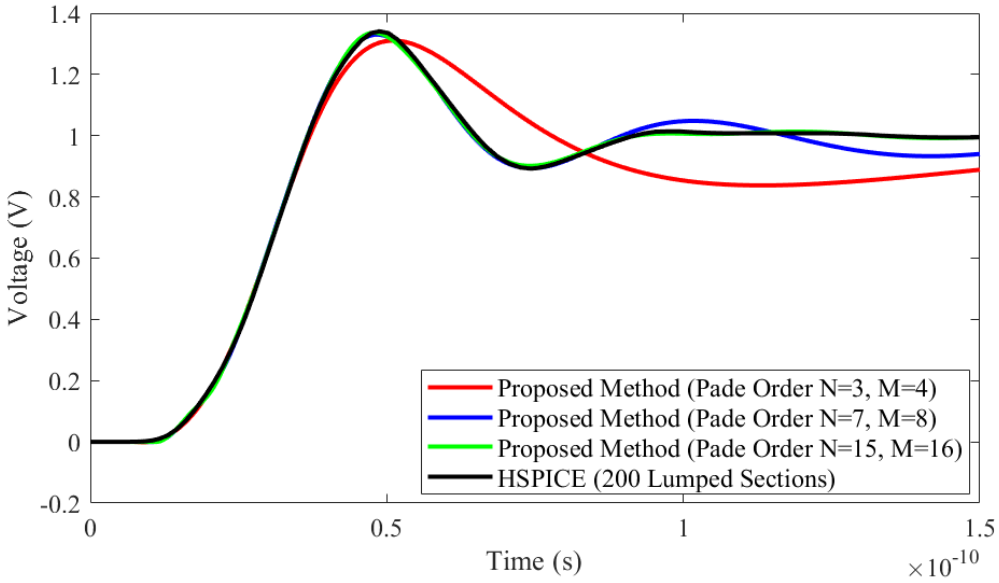


(b)

Figure 4. 24: The far end time domain response at node N_7 for Example 3 with unit rising ramp input of 0.1 ns rise time, l_x = 0.2 mm, R_s = 10 Ω, C_x= 20 fF (a) Comparison of proposed method Padé order (N = 1, M = 2), HSPICE and two-pole model [25] (b) Comparison of proposed method Padé orders (N = 3, M = 4), (N = 7, M = 8) and (N = 15, M = 16), respectively, against HSPICE



(a)



(b)

Figure 4. 25: The far end time domain response at node N_7 for Example 3 with unit rising ramp input of 0.025 ns rise time, l_x = 0.2 mm, R_s = 10 Ω, C_x = 20 fF (a) Comparison of proposed method Padé order (N = 1, M = 2), HSPICE and two-pole model [25] (b) Comparison of proposed method Padé orders (N = 3, M = 4), (N = 7, M = 8) and (N = 15, M = 16), respectively, against HSPICE

Table 4. 28: Comparison of 10%, 50% and 90% delays at node N_7 using the proposed algorithm and two-pole model [25] for Example 3 with unit rising ramp input of 0.1 ns rise time

l_x	R_s	C_x	HSPICE			Two-Pole Model [25]			Proposed Method (Padé Order N, M)											
									(1, 2)			(3, 4)			(7, 8)			(15, 16)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
(mm)	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
0.2	10	20	25.97	58.62	100.19	24.60	59.40	100.40	25.35	61.33	115.75	25.85	57.77	102.37	25.91	59.23	99.42	25.95	59.28	100.16
	10	500	57.22	111.88	183.91	52.80	112.80	192.90	55.35	120.21	223.31	57.30	112.16	179.23	57.31	111.93	184.86	57.21	111.92	184.17
	30	20	31.18	77.32	120.57	30.90	77.40	123.90	31.89	78.82	149.32	31.36	76.58	126.45	31.22	77.00	122.52	31.18	76.87	120.26
	30	100	43.15	96.96	178.71	41.10	97.50	175.80	42.79	102.13	197.81	43.07	96.25	169.58	43.09	97.65	181.06	43.15	97.66	178.79
1	10	20	79.65	124.16	157.69	62.80	122.00	175.60	73.50	128.43	205.58	80.99	124.24	159.62	79.75	124.40	157.88	79.68	124.45	157.56
	10	100	86.29	138.73	180.61	69.60	135.60	202.00	80.91	144.64	235.05	88.15	139.00	182.00	85.92	138.81	179.30	85.63	139.08	181.31
	20	20	86.75	138.71	191.99	68.40	139.80	234.60	79.10	150.03	205.80	85.83	138.4	196.25	85.13	138.24	191.48	85.57	138.75	190.73
	20	100	93.27	156.30	232.37	76.20	157.80	280.80	87.50	170.57	309.42	94.15	156.24	231.53	93.15	154.76	232.22	93.21	155.95	231.35
Average Error (%)						12.3	1.1	9.5	4.7	5.5	21.5	0.87	0.49	2.4	0.40	0.47	0.67	0.29	0.40	0.25
Maximum Error (%)						21.2	2.3	22.2	8.8	9.1	33.2	2.2	1.5	5.1	1.9	1.0	1.6	1.4	1.1	0.66

Table 4. 29: Comparison of 10%, 50% and 90% delays at node N_7 using the proposed algorithm and two-pole model [25] for Example 3 with unit rising ramp input of 0.025 ns rise time

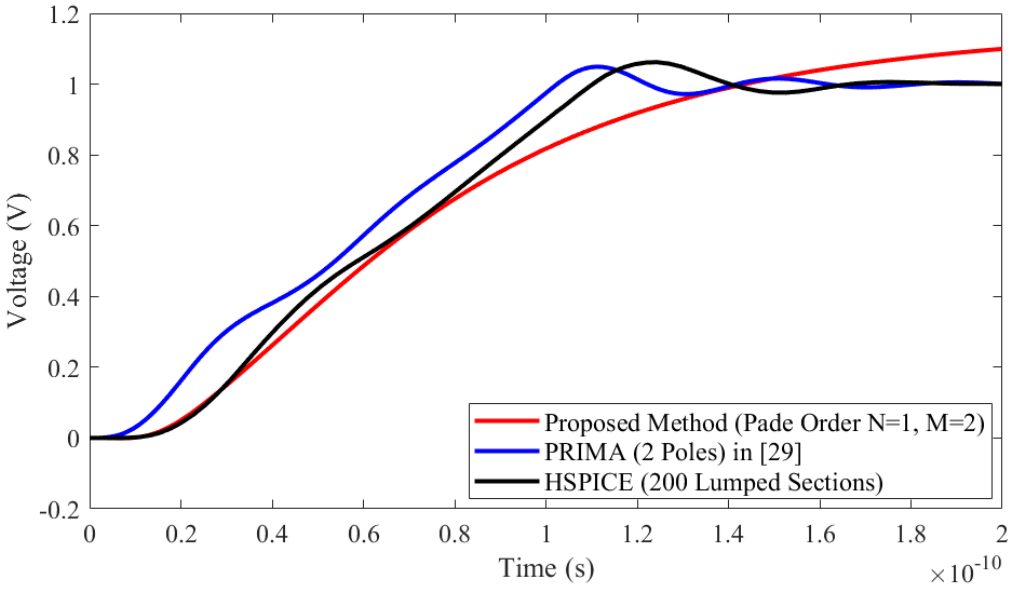
l_x	R_s	C_x	HSPICE			Two-Pole Model [25]			Proposed Method (Padé order N, M)											
									(1, 2)			(3, 4)			(7, 8)			(15, 16)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
(mm)	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
0.2	10	20	17.44	27.86	34.86	14.10	27.00	37.30	16.27	28.00	43.49	17.72	27.72	34.98	17.28	27.77	34.76	17.31	27.78	34.82
	10	500	34.99	73.28	121.34	30.40	71.20	147.60	34.43	77.70	150.93	35.11	72.59	123.32	34.95	72.56	121.12	34.98	72.38	121.20
	30	20	20.78	34.30	77.85	16.50	35.50	71.00	18.86	38.42	73.89	20.18	34.38	61.42	20.40	34.21	77.31	20.62	34.28	76.97
	30	100	26.79	50.33	121.62	22.20	53.70	130.50	25.28	55.60	123.39	26.42	50.34	153.47	26.91	50.38	121.25	26.74	50.05	121.27
1	10	20	63.82	87.30	111.92	38.40	84.00	132.60	55.26	90.76	141.82	62.88	89.38	113.89	64.13	87.85	113.93	63.69	88.89	110.94
	10	100	67.96	107.97	132.16	43.00	97.50	159.00	60.83	107.30	169.97	66.72	102.03	135.93	68.10	105.63	135.33	67.92	108.53	132.24
	20	20	65.67	100.40	147.63	41.40	99.60	190.80	58.83	108.83	203.02	65.10	99.23	140.28	65.44	100.22	143.16	65.55	101.64	146.45
	20	100	69.90	122.77	189.09	46.80	118.20	238.20	65.35	127.76	236.59	69.60	116.05	177.17	69.63	118.62	178.1	69.74	121.91	189.18
Average Error (%)						27.1	4.2	17.3	8.0	5.8	21.7	1.4	2.0	8.2	0.59	1.0	1.8	0.30	0.80	0.43
Maximum Error (%)						39.8	9.7	29.2	13.4	12.0	37.5	2.9	5.5	26.2	1.8	3.4	5.8	0.77	1.8	1.1

4.2.3.2 Comparing the Proposed Algorithm to PRIMA

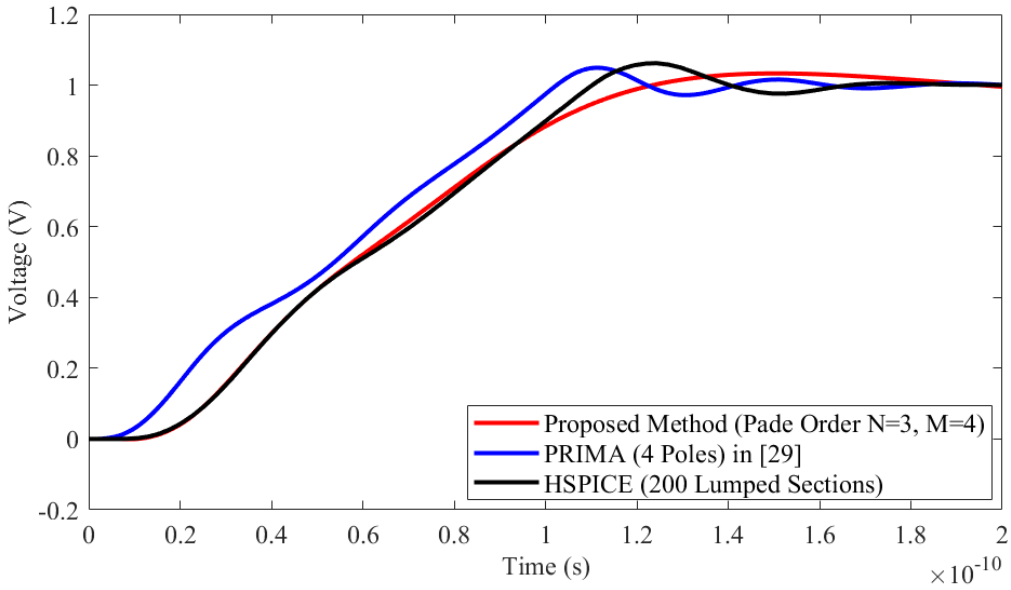
As discussed in Section 4.1.3, the given example here is a six-port network, the approximation orders of 2, 4, 8 and 16 for PRIMA [29] are considered to compare with the Padé orders of (N = 1, M = 2), (N = 3, M = 4), (N = 7, M = 8) and (N = 15, M = 16) for the proposed algorithm. One moment is matched by approximation order 2, which matches the same moment as approximation order 4. As well two and three moments are matched by approximation orders 8 and 16, respectively. Figure 4.26 (a) and Figure 4.27 (a) show the transient responses obtained from the NILT with Padé order (N = 1, M = 2), HSPICE and the PRIMA with approximation order of 2, respectively, for l_x = 0.2 mm, R_s = 10 Ω, C_x= 20 fF when the rise times of ramp input are 0.1 ns and 0.025 ns. Figure 4.26 (b)-(d) and Figure 4.27 (b)-(d) show the comparison of responses of the higher order approximations between the NILT (Padé orders (N = 3, M = 4), (N = 7, M = 8) and (N = 15, M = 16), respectively) and the PRIMA (approximation orders 4, 8 and 16, respectively). From the Figures 4.26 and 4.27, the proposed algorithm provides more accurate estimations for 10%, 50% and 90% delay compared to PRIMA with the unit rising ramp input, while the PRIMA is more accurate at the steady state. Once again, the proposed algorithm can improve the accuracy of delay estimation by increasing the Padé order. Both the proposed algorithm and PRIMA match the HSPICE well when the highest approximation order is considered in this example. The delay results at the node N_7 are calculated using the proposed algorithm and PRIMA, respectively and compared to those of HSPICE to verify the accuracy of the proposed algorithm (Tables 4.30 and 4.31).

The approximation orders of 2 and 4 for PRIMA generate the same results of 10%, 50% and 90% delay since they both can match only one moment (Tables 4.30 and 4.31). For the proposed algorithm, the average errors of the 10%, 50% and 90% delay estimations are decreased obviously when the Padé order is increased from (N = 1, M = 2) to (N = 3, M = 4). For PRIMA, when the approximation order is increased to 8, the accuracies of delay estimations are improved significantly compared to the lower approximation order 2 or 4. Further increasing the approximation order to 16 for PRIMA, the accuracies of the delay estimations keep unchanged comparing with approximation order of 8 though the

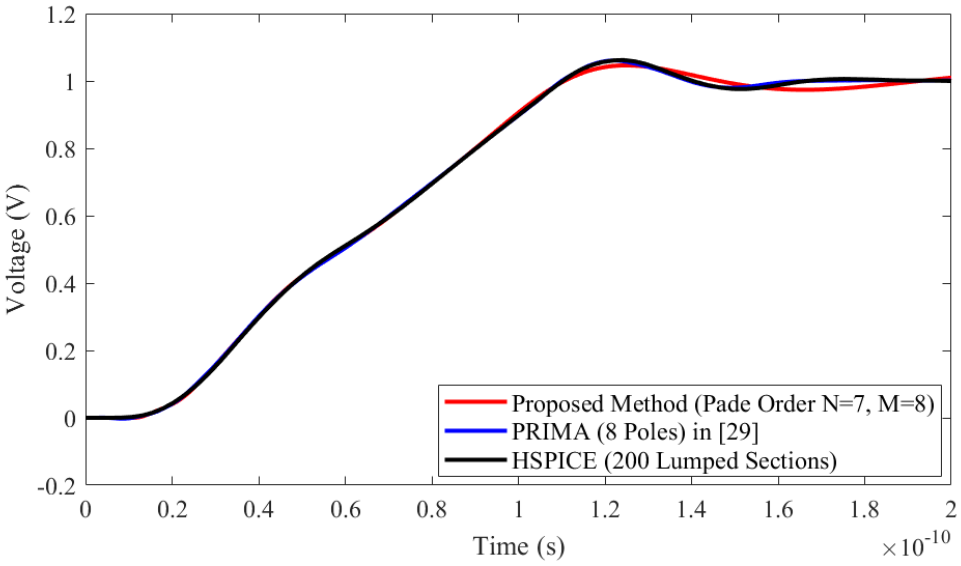
accuracy of steady state is improved. As a comparison, the proposed algorithm improves the accuracies of delay estimations obviously when the Padé order increases. For Padé order (N = 15, M = 16), the average errors of 10%, 50% and 90% delay are the lowest compared with lower Padé orders.



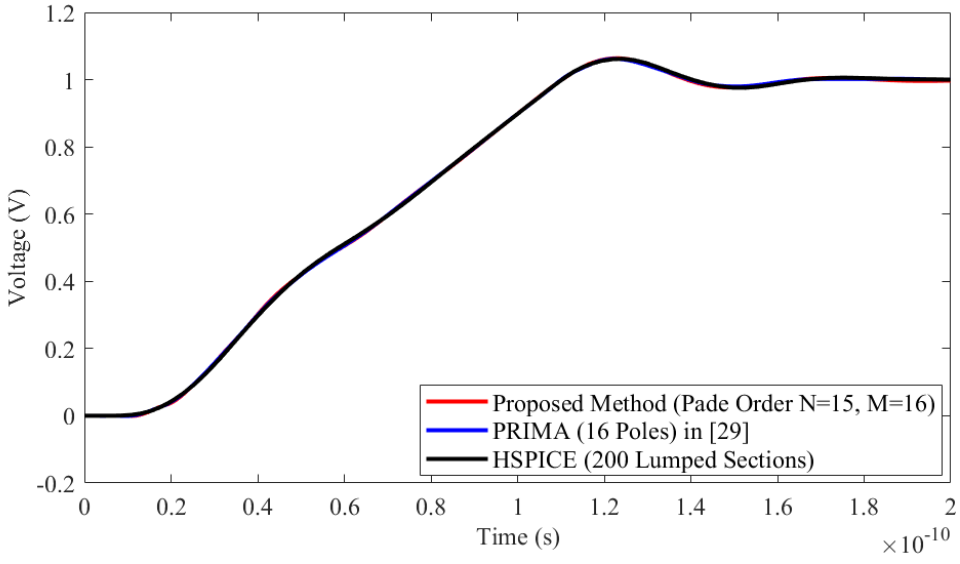
(a)



(b)



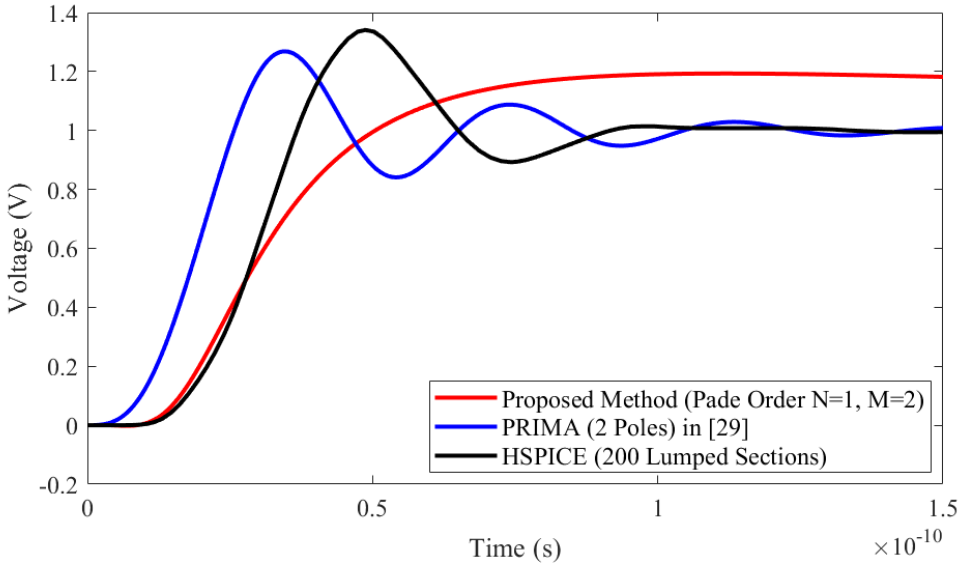
(c)



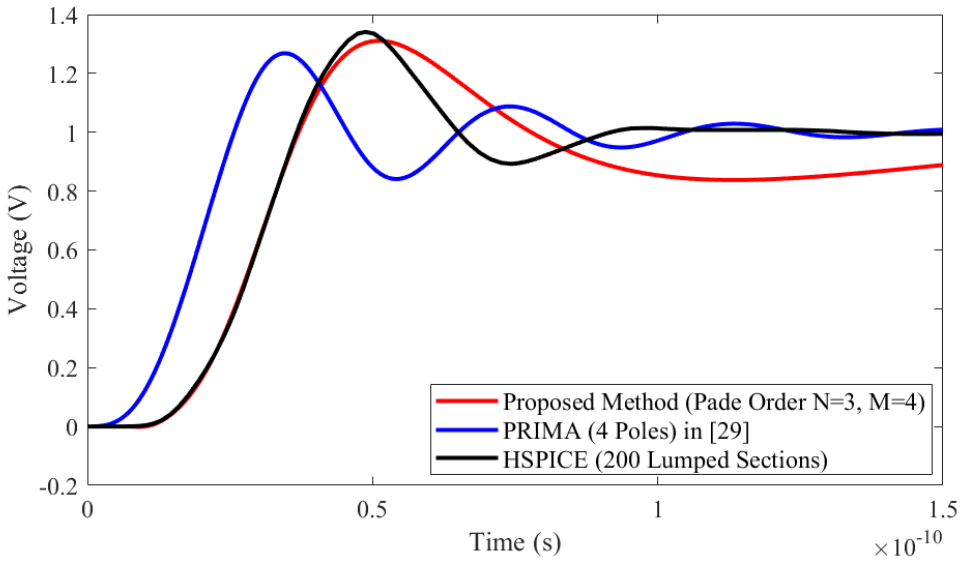
(d)

Figure 4. 26: The far end time domain response at node N_7 for Example 3 with unit rising ramp input of 0.1 ns rise time, l_x = 0.2 mm, R_s = 10 Ω, C_x= 20 fF (a) Comparison of proposed method Padé order (N = 1, M = 2), HSPICE and PRIAM (2 poles) [29] (b) Comparison of proposed method Padé order (N = 3, M = 4), HSPICE and PRIAM (4 poles) [29] (c) Comparison of proposed method Padé order (N = 7, M = 8), HSPICE and PRIAM (8 poles) [29] (d) Comparison of proposed method Padé order (N = 15, M = 16), HSPICE and PRIAM (16 poles) [29]

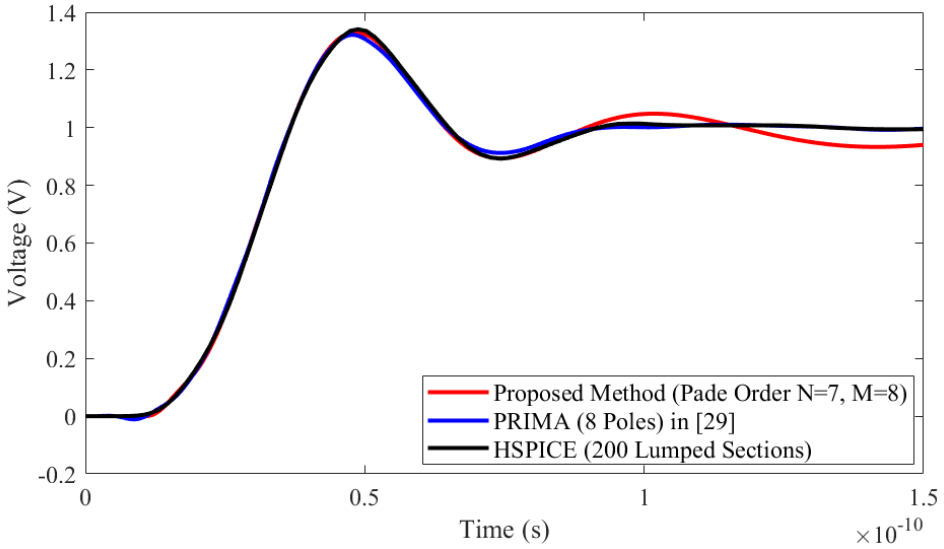
The average run times solving nodes N_1 to N_9 at 500 time points using both the proposed algorithm and PRIMA are calculated and compared, which are listed in Table 4.32 for the case when l_x = 0.2 mm. As seen from Table 4.32, the proposed method requires less run time than PRIMA. For the same approximation order of 16, the average run times for PRIMA and the proposed method are 873.15 ms and 100.16 ms for rise time of 0.1 ns, and 917.21 ms and 61.02 ms for rise time of 0.025 ns, respectively. The results show that the proposed algorithm is about 9 to 15 times faster than the PRIMA for the highest approximation order in this example.



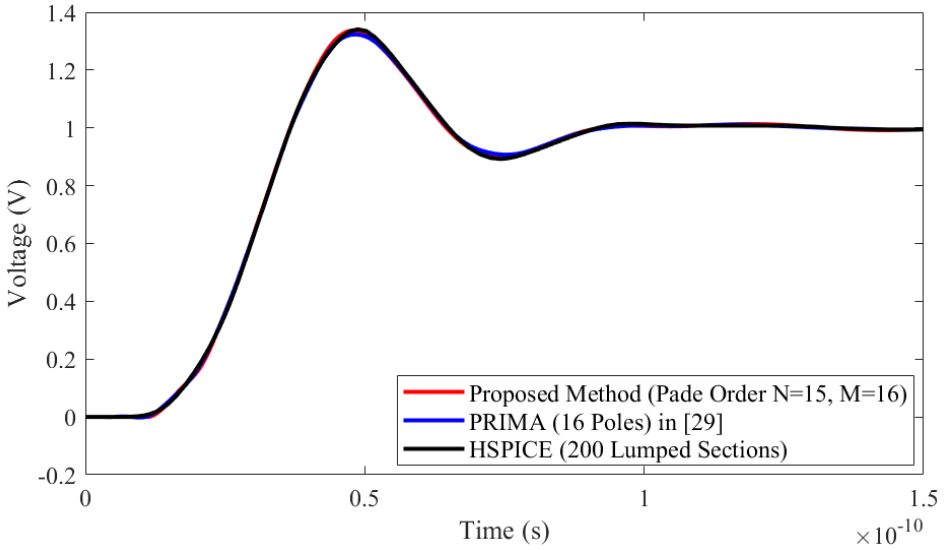
(a)



(b)



(c)



(d)

Figure 4. 27: The far end time domain response at node N_7 for Example 3 with unit rising ramp input of 0.025 ns rise time, l_x = 0.2 mm, R_s = 10 Ω, C_x = 20 fF (a) Comparison of proposed method Padé order (N = 1, M = 2), HSPICE and PRIAM (2 poles) [29] (b) Comparison of proposed method Padé order (N = 3, M = 4), HSPICE and PRIAM (4 poles) [29] (c) Comparison of proposed method Padé order (N = 7, M = 8), HSPICE and PRIAM (8 poles) [29] (d) Comparison of proposed method Padé order (N = 15, M = 16), HSPICE and PRIAM (16 poles) [29]

Table 4. 30: Comparisons of 10%, 50% and 90% delays at node N_7 using the proposed method and PRIMA [29] for Example 3 with unit rising ramp input of 0.1 ns rise time

l_x	R_s	C_x	HSPICE			PRIMA [29] (Approximation Order)						Proposed Method (Padé order N, M)					
						2			4			(1, 2)			(3, 4)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
(mm)	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
0.2	10	20	25.97	58.62	100.19	16.02	53.81	92.90	16.02	53.81	92.90	25.35	61.33	115.75	25.85	57.77	102.37
	10	500	57.22	111.88	183.91	53.60	107.12	169.78	53.60	107.12	169.78	55.35	120.21	223.31	57.30	112.16	179.23
	30	20	31.18	77.32	120.57	18.36	56.72	96.68	18.36	56.72	96.68	31.89	78.82	149.32	31.36	76.58	126.45
	30	100	43.15	96.96	178.71	34.74	81.90	130.29	34.74	81.90	130.29	42.79	102.13	197.81	43.07	96.25	169.58
1	10	20	79.65	124.16	157.69	25.30	51.70	94.90	25.30	51.70	94.90	73.50	128.43	205.58	80.99	124.24	159.62
	10	100	86.29	138.73	180.61	44.61	84.33	113.10	44.61	84.33	113.10	80.91	144.64	235.05	88.15	139.00	182.00
	20	20	86.75	138.71	191.99	25.95	54.30	96.27	25.95	54.30	96.27	79.10	150.03	205.80	85.83	138.4	196.25
	20	100	93.27	156.30	232.37	46.98	90.48	123.84	46.98	90.48	123.84	87.50	170.57	309.42	94.15	156.24	231.53
Average Error (%)						42.7	31.9	29.5	42.7	31.9	29.5	4.7	5.5	21.5	0.87	0.49	2.4
Maximum Error (%)						70.1	60.9	49.9	70.1	60.9	49.9	8.8	9.1	33.2	2.2	1.5	5.1

Table 4.30 (Cont'd.):

l_x	R_s	C_x	HSPICE			PRIMA [29] (Approximation Order)						Proposed Method (Padé order N, M)					
						8			16			(7, 8)			(15, 16)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
(mm)	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
0.2	10	20	25.97	58.62	100.19	25.64	59.49	100.26	25.64	59.49	100.26	25.91	59.23	99.42	25.95	59.28	100.16
	10	500	57.22	111.88	183.91	57.00	111.80	184.64	57.00	111.80	184.64	57.31	111.93	184.86	57.21	111.92	184.17
	30	20	31.18	77.32	120.57	31.23	76.89	120.51	31.23	76.89	120.51	31.22	77.00	122.52	31.18	76.87	120.26
	30	100	43.15	96.96	178.71	42.96	97.55	178.41	42.96	97.55	178.41	43.09	97.65	181.06	43.15	97.66	178.79
1	10	20	79.65	124.16	157.69	80.92	124.48	157.22	80.92	124.48	157.22	79.75	124.40	157.88	79.68	124.45	157.56
	10	100	86.29	138.73	180.61	88.26	137.73	179.82	88.26	137.73	179.82	85.92	138.81	179.30	85.63	139.08	181.31
	20	20	86.75	138.71	191.99	85.41	137.76	192.42	85.41	137.76	192.42	85.13	138.24	191.48	85.57	138.75	190.73
	20	100	93.27	156.30	232.37	94.32	154.83	235.02	94.32	154.83	235.02	93.15	154.76	232.22	93.21	155.95	231.35
Average Error (%)						1.1	0.67	0.35	1.1	0.67	0.35	0.40	0.47	0.67	0.29	0.40	0.25
Maximum Error (%)						2.3	1.5	1.1	2.3	1.5	1.1	1.9	1.0	1.6	1.4	1.1	0.66

Table 4. 31: Comparisons of 10%, 50% and 90% delays at node N_7 using the proposed method and PRIMA [29] for Example 3 with unit rising ramp input of 0.025 ns rise time

l_x	R_s	C_x	HSPICE			PRIMA [29] (Approximation Order)						Proposed Method (Padé order N, M)					
						2			4			(1, 2)			(3, 4)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
(mm)	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
0.2	10	20	17.44	27.86	34.86	9.38	17.70	23.99	9.38	17.70	23.99	16.27	28.00	43.49	17.72	27.72	34.98
	10	500	34.99	73.28	121.34	31.96	68.06	111.94	31.96	68.06	111.94	34.43	77.70	150.93	35.11	72.59	123.32
	30	20	20.78	34.30	77.85	10.35	20.45	29.11	10.35	20.45	29.11	18.86	38.42	73.89	20.18	34.38	61.42
	30	100	26.79	50.33	121.62	19.97	39.78	88.98	19.97	39.78	88.98	25.28	55.60	123.39	26.42	50.34	153.47
1	10	20	63.82	87.30	111.92	15.27	27.45	35.43	15.27	27.45	35.43	55.26	90.76	141.82	62.88	89.38	113.89
	10	100	67.96	107.97	132.16	27.45	51.58	70.63	27.45	51.58	70.63	60.83	107.30	169.97	66.72	102.03	135.93
	20	20	65.67	100.40	147.63	15.63	28.35	36.99	15.63	28.35	36.99	58.83	108.83	203.02	65.10	99.23	140.28
	20	100	69.90	122.77	189.09	28.50	55.53	79.05	28.50	55.53	79.05	65.35	127.76	236.59	69.60	116.05	177.17
Average Error (%)						50.2	44.0	47.1	50.2	44.0	47.1	8.0	5.8	21.7	1.4	2.0	8.2
Maximum Error (%)						76.2	71.8	74.9	76.2	71.8	74.9	13.4	12.0	37.5	2.9	5.5	26.2

Table 4.31 (Cont'd.):

l_x	R_s	C_x	HSPICE			PRIMA [29] (Approximation Order)						Proposed Method (Padé order N, M)					
						8			16			(7, 8)			(15, 16)		
			10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay	10% Delay	50% Delay	90% Delay
(mm)	(Ω)	(fF)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)	(ps)
0.2	10	20	17.44	27.86	34.86	17.63	27.62	34.66	17.63	27.62	34.66	17.28	27.77	34.76	17.31	27.78	34.82
	10	500	34.99	73.28	121.34	34.62	72.42	122.32	34.62	72.42	122.32	34.95	72.56	121.12	34.98	72.38	121.20
	30	20	20.78	34.30	77.85	20.06	34.08	74.61	20.06	34.08	74.61	20.40	34.21	77.31	20.62	34.28	76.97
	30	100	26.79	50.33	121.62	26.43	50.16	121.69	26.43	50.16	121.69	26.91	50.38	121.25	26.74	50.05	121.27
1	10	20	63.82	87.30	111.92	60.99	87.57	112.59	60.99	87.57	112.59	64.13	87.85	113.93	63.69	88.89	110.94
	10	100	67.96	107.97	132.16	73.55	114.77	149.85	73.55	114.77	149.85	68.10	105.63	135.33	67.92	108.53	132.24
	20	20	65.67	100.40	147.63	62.49	97.71	145.02	62.49	97.71	145.02	65.44	100.22	143.16	65.55	101.64	146.45
	20	100	69.90	122.77	189.09	63.15	116.97	181.38	63.15	116.97	181.38	69.63	118.62	178.1	69.74	121.91	189.18
Average Error (%)						4.3	2.1	3.2	4.3	2.1	3.2	0.59	1.0	1.8	0.30	0.80	0.43
Maximum Error (%)						9.7	6.3	13.4	9.7	6.3	13.4	1.8	3.4	5.8	0.77	1.8	1.1

Table 4. 32: Run time comparison between the proposed method and PRIMA [29] for Example 3

Rise Time (ns)	PRIMA [29]		Proposed Method		Speed Up
	Approximation Order	Run Time (ms)	Padé Order	Run Time (ms)	
0.1	2	756.73	(1, 2)	19.81	38
	4	756.73	(3, 4)	30.99	24
	8	836.90	(7, 8)	52.86	16
	16	873.15	(15, 16)	100.16	9
0.025	2	813.53	(1, 2)	12.98	63
	4	813.53	(3, 4)	20.05	41
	8	896.86	(7, 8)	32.09	28
	16	917.21	(15, 16)	61.02	15

4.2.4 Summary of the Results

For the unit rising ramp input, all the two-pole model [25], PRIMA [29] and the proposed algorithm provide more accurate estimations for 10%, 50% and 90% delays compared to unit step input case. Further, the two-pole model provides better estimates for the 50% and 90% delays, while NILT Padé order (N = 1, M = 2) provides better estimates for the 10% delay. Typically, as the rise time of the input signal decreases and approaches the unit step response, NILT Padé order (N = 1, M= 2) provides better accuracy than the two-poles model and as the rise time increases the two-pole model becomes more accurate compared to the Padé order (N = 1, M = 2) approximation. From the average run times comparison, the two-pole model requires less run time compared to the proposed algorithm for Padé order (N = 1, M = 2), however, the two-pole model may suffer numerical stability problem in practice due to the large inductance value of transmission line. Conversely, the proposed method is suitable for inductance dominant cases and is always numerical stable. As well the proposed algorithm can visibly improve the accuracy of delay estimations by increasing the Padé orders without numerical issue experienced by the two-pole model.

When comparing to PRIMA, the proposed method presents advantages in both accuracy and efficiency. PRIMA can improve the accuracy of the delays significantly when the approximation order is increased from 2 to 8. However, the accuracy of the delays keeps unchanged when the approximation order is further increased from 8 to 16 though the accuracy of steady state is improved obviously when compare to the proposed algorithm. Since more sections of lumped RLC are required to model a transmission line, and enough number of lumped RLC sections is needed to match the required moments, which enlarges the size of matrices in application of PRIMA. Huge size of matrices results in expensively computational cost, which lowers the efficiency of PRIMA.

4.3 Conclusion

In this work, a new analytical delay model based on the NILT is proposed for on-chip RLC interconnect networks. Using the proposed algorithm, the 10%, 50% and 90% delay estimations are calculated by selecting different Padé orders. Even the accuracies of the delay estimations can be greatly improved by increasing the Padé orders without causing numerical stability issues. With the unit step and unit rising ramp inputs for different rise time, the proposed algorithm is compared to the conventional two-pole model [25] and the advanced moment matching based PRIMA [29] in respect of accuracy and efficiency.

For unit step input, the proposed algorithm is capable of providing more accurate estimations for delays compared to the two-pole model and PRIMA. The proposed algorithm can significantly improve the accuracies of the delay estimations compared to the two-pole model and PRIMA. Conversely, increasing the number of poles by using a higher order Maclaurin series to approximate the cosh and sinh terms following the steps of [25] results in unstable three-pole models. For PRIMA, further increasing approximation order to a higher level, the accuracy of delays is not improved though the accuracy of steady state is improved. For unit rising ramp input with different rise time, the two-pole model is more accurate than the proposed algorithm when estimating 50% and 90% delays, while the proposed algorithm is more accurate when calculating 10% delay compared to the two-pole model. As well, the proposed algorithm is more accurate than PRIMA when estimating delays at the early stage.

From the average run times comparison, the two-pole model runs faster than the proposed method for Padé order ($N = 1$, $M = 2$) while the two-pole model may not always be stable because the large inductance value of the transmission line is likely to cause instability when matching moments. When comparing the proposed algorithm to PRIMA, the proposed algorithm requires less run time since the size of matrices used in the proposed algorithm is much smaller than that of used in PRIMA. By detailed analysis, it is evident that the proposed method has advantages over the conventional two-pole model and advanced moment matching based PRIMA in terms of accuracy, numerical stability and efficiency.

Chapter 5

Conclusions and Future Work

5.1 Summary

An analytic delay model has been presented for transient analysis of on-chip *RLC* distributed interconnects widely used in VLSI circuits. The proposed method is based on the numerical inversion of Laplace transform, which can exactly inverse a certain number of terms of the Taylor series of the time domain responses using rational Padé approximation, and avoid the complicated tasks in finding poles and residues. In this thesis, the major advantages of the proposed method have been demonstrated as follows.

1. The frequency domain transfer function for any node of interconnect structure is obtained exactly in term of the interconnect parameters such as the per-unit-length parameters and equivalent input impedance. This fact provides substantial advantage for the proposed method over other existing techniques, which depend on macromodeling algorithms or moment matching techniques to obtain an approximation for the transfer function of interconnects. In addition, the equivalent circuit models describing the practical interconnect structures are presented for transfer function derivation and transient response analysis of interconnect.

2. For the NILT based proposed method, the time domain response of the interconnect under consideration can be obtained with arbitrary accuracy by selecting various approximation orders. The proposed method can solve the problems occurred in conventional implicit LMS (Linear Multi-Step) methods, in which the order greater than two are not absolutely stable and it is challenging to use higher order LMS methods to derive a time domain expression similar to the proposed method since the time interval can affect the stability of the function. For the proposed method, the accuracy of the calculated results can be effectively improved by increasing approximation orders without causing numerically stable issues.

3. Though the number of poles and residues increase obviously as the approximation

order increase, the proposed method only needs to compute the frequency domain function at preassigned complex points, which avoids the complicated and cumbersome finding tasks for poles and residues. The proposed method is thus applicable for analyzing time domain response of interconnects, which contains infinite number of poles due to their distributive characteristics.

4. The efficiency of NILT stems from the fact that it directly uses the frequency solution of interconnects without requiring macromodeling approximations, in which an interconnect is modeled by dividing into number of lumped *RLC* sections. This adds additional voltage and current variables to the MNA circuit equations, which increases computational complexity of the time domain response. Conversely, the matrices required in the proposed method are constructed by interconnect's length and per-unit-length parameters, which determines the relatively small matrix size compared with the implicit LMS integration formulas such as Backward Euler, Trapezoidal Rule and Gear's method used in SPICE. This fact significantly improves the computational efficiency of the proposed method.

5. Due to the criterion of approximation order selection, all the poles of the rational Padé approximation used in the proposed method are located only in right half plane and produces integration formulas that are always stable. Furthermore, the poles of the rational Padé approximation are mostly complex conjugate pairs, the resulting real time function is evaluated using only the poles located in the upper half plane, which reduces the computation of the time function in half.

6. The proposed algorithm provides a mechanism to increase the accuracy of the model for electrically long *RLC* interconnect circuits. Furthermore, the higher order time functions obtained by NILT can be solved in parallel using multi-core processing techniques because the independence of each summation term in the proposed method.

5.2 Future Work

Some suggestions for further research based on the results of this thesis are described in

this section.

1. Stepping algorithm for on-chip interconnect networks

   The proposed method in this thesis is extremely simple and precise to apply for small times but has the disadvantage that accuracy decreases as time increases. It has been pointed out in [45] that smaller denominator degree *M* of the rational Padé approximation of (3.30) provides more accurate results for small time *t*. This issue can be explained by the fact that the values of residues K_i' of (3.36) grow rapidly with *M* and roundoff errors increase the overall error. For large times, the higher number of matched terms of the Taylor expansion make higher *N*, *M* of (3.30) desirable. Thus, stepping algorithm application becomes necessary for large times, by which it is possible to use the proposed method with small time intervals, where the accuracy is excellent, and reset the problem after a small step so that in the next evaluation the previous result is considered as the initial point for the new step, as in the numerical integration of differential equations. Applying stepping algorithm to the proposed method of this thesis can obtain high accuracy even for large times.

2. The NILT based waveform relaxation algorithm for nonlinear interconnects

   The proposed method in this thesis is only applicable for the linear interconnect systems but not for nonlinear case. Waveform relaxation (WR) technique provides a powerful tool for analyzing interconnect system with arbitrary nonlinearity including strong nonlinearity, which can accurately present the effects of the nonlinear elements on the performance of the overall system by solving subsystems [70], [71]. Using WR to decompose the interconnect from its terminations so that the nonlinear and the linear subsystems are independent and can be analyzed efficiently. Employing the desired overlapping technique to provide a better WR convergence between linear and nonlinear subsystems. For the linear subsystems, the proposed method of this thesis is used to obtain the time domain solution with high accuracy, while the numerical integration algorithm, such as trapezoidal rule, is considered to convert the nonlinear ODEs into algebraic equations, which is then

solved using Newton Raphson algorithm for the nonlinear subnetwork. As a result, accurate time domain solution for the nonlinear interconnect system is possible.

3. The NILT based waveform relaxation algorithm for large interconnect systems

   As mentioned in the previous section, the proposed method in this thesis is only suitable for linear interconnect system. For a complex and large interconnect system containing multiple levels of interconnects, the WR method based on longitudinal partitioning (LP) scheme is used to partition the original large interconnect system into multiple levels and each level contains multiple subsystems and even each subsystem is further decomposed into an interconnection of smaller subsystems [70], [71]. This, thus, produces a hierarchy of decomposition level. Multiple subsystems per level generated by the WR algorithm permits the use of a large number of parallel processors, which results in computation time savings in implementation procedure. In addition, the proposed method of this thesis and trapezoidal rule are used for linear and nonlinear subsystems, respectively, to obtain the accurate estimation of the time domain response for the given large interconnect system.

Reference

[1] M.S. Ullah and M. H. Chowdhury, “Analytical models of high–speed RLC interconnect delay for complex and real poles,” *IEEE Trans. Very Large Scale Integr. (VLSI) Syst*., vol. 25, no. 6, pp. 1831-1841, Jun. 2017.

[2] S. Jadav, “High speed RLC equivalent RC delay model for global VLSI interconnects,” *Analog Integrated Circuits and Signal Processing*, vol. 100, no. 1, pp.109-117, Jul. 2019.

[3] H.B. Bakoglu, *Circuit, interconnections and packaging for VLSI*. Reading, MA: Addison-Wesley, 1990.

[4] J. Rosenfeld and E. G. Friedman, “Design methodology of global resonant H-tree clock distribution networks,” *IEEE Trans. VLSI Systems*, vol. 15, no. 2, pp. 135-148, Feb. 2007.

[5] F. S. H. Lori, M. S. Hosen, A. Menshov, and M. Shafieipour, “New higher order method of moments for accurate inductance extraction in transmission lines of complex cross sections,” *IEEE Transactions on Microwave Theory and Techniques*, vol. 65, no. 12, pp. 5104-5112, Dec. 2017.

[6] Y. I. Ismail and E. G. Friedman, *On-Chip Inductance in High Speed Integrated Circuits*. Norwell, MA: Kluwer, 2001.

[7] M. Celik, L. Pileggi, and A. Odabasioglu, *IC Interconnect Analysis*. Norwell, MA: Kluwer, 2002.

[8] S. Roy and A. Dounavis, “Parallel transient simulation of package board power distribution networks based on a 2-D overlapping partitioning methodology,” *IEEE Transactions on Components Packaging and Manufacturing Technology*, vol. 3, no. 12, pp. 2101-2112, Dec. 2013.

[9] E. Rasekh and A. Dounavis, “A multidimensional Krylov reduction technique with constraint variables to model nonlinear distributed networks,” *IEEE Transactions on Advanced Packaging*, vol. 33, no. 3, pp. 738-746, 2010.

[10] M. S. Zhang, “Dynamic charge exchange and delivery of the high-speed power delivery networks,” *IEEE Electromagnetic Compatibility Magazine*, vol. 6, no. 2, pp. 89-99, Aug. 2017.

[11] T. Sakurai, “Approximation of wiring delay in MOSFET LSI,” *J. Solid-State Circuits*, vol. 18, no. 4, pp. 418-426, Aug. 1983.

[12] G. Y. Yacoub, H. Pham, and E. G. Friedman, “A system for critical path analysis based on back annotation and distributed interconnect impedance models,” *Microelectron. J.*, vol. 18, no. 3, pp. 21-30, June 1988.

[13] A. Deutsch et al., “When are transmission-line effects important for on-chip interconnections,” *IEEE Trans. Microwave Theory Tech.*, vol. 45, no. 10, pp. 1836-1997, Oct. 1997.

[14] J. C. Liao, O. A. Palusinski, and J. L. Prince, “Transmission line simulator as a basic component of CAD system for VLSI interconnections,” *Proc. Int. Phoenix Conf. on Computers and Communications*, Mar. 1990, pp. 21-24.

[15] A. Dounavis, V. A. Pothiwala, “Passive closed form transmission line macromodel using method of characteristics,” *IEEE Trans. Adv. Packag*, vol. 31, no. 1, pp. 190-202, Feb. 2008.

[16] A. B. Kahng and S. Muddu, “An analytic delay model for RLC interconnects,” *IEEE Trans. CAD of Integrated Circuits and Syst*, vol. 16, no. 12, pp. 1507-1514, Dec. 1997.

[17] K. Bannerjee and A. Mehrotra, “Analysis of on-chip inductance effects for distributed RLC interconnects,” *IEEE Trans. CAD of Integrated Circuits and Syst.*, vol. 21, no. 8, pp. 904-915, Aug. 2002.

[18] D. B. Kuznetsov, J. E. Schutt-Ainé, “Optimal transient simulation of transmission lines,” *IEEE Trans. Circuits Syst. I*, vol. 43, pp. 111-121, Feb. 1996.

[19] N. M. Nakhla, A. Dounavis, R. Achar, and M. S. Nakhla, “DEPACT: delay extraction-based passive compact transmission-line macromodeling algorithm,” *IEEE Transactions on Components, Packaging, and Manufacturing Technology*, vol. 28, no.1, pp.13-23, Feb. 2005.

[20] Y. Eo, S. Shin, W. R. Eisenstadt, and J. Shim, “Generalized traveling-wave-based waveform approximation technique for the efficient signal integrity verification technique of multicoupled transmission lines,” *IEEE Trans. Comput. -Aided Design Integr. Circuits Syst*., vol. 21, no. 12, pp. 1489-1497, Dec. 2002.

[21] W. C. Elmore, “The transient response of damped linear networks with particular regard to wideband amplifiers,” *J. Appl. Phys*., vol. 19, no. 1, pp. 55-63, Jan. 1948.

[22] T. Sakurai, “Closed-form expressions for interconnection delay, coupling, and crosstalk in VLSIs,” *IEEE Trans. Electron Devices*, vol. 40, no. 1, pp. 118-124, Jan. 1993.

[23] Y. I. Ismail and E. G. Friedman, “Effects of inductance on the propagation delay and repeater insertion in VLSI circuits,” *IEEE Trans. Very Large Scale Integr. (VLSI) Syst*., vol. 8, no. 2, pp. 195-206, Apr. 2000.

[24] Y. I. Ismail, E. G. Friedman, and J. L. Neves, “Equivalent Elmore delay for RLC trees,” *IEEE Trans. Comput. -Aided Design Integr. Circuits Syst*., vol. 19, no. 1, pp. 83-97, Jan. 2000.

[25] X. C. Li, J. F. Mao, and H. F. Huang, “Accurate analysis of interconnect trees with distributed RLC model and moment matching,” *IEEE Trans. Microw. Theory Tech*., vol. 52, no. 9, pp. 2199-2206, Sep. 2004.

[26] L. T. Pillage and R. A. Rohrer, “Asymptotic waveform evaluation for timing analysis,” *IEEE Trans. Computer-Aided Design*, vol. 9, no. 4, pp. 352-366, Apr. 1990.

[27] D. F. Anastasakis, N. Gopal, S. Y. Kim, and L.T. Pillage, “Enhancing the stability of asymptotic waveform evaluation for digital interconnect circuit applications,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol.13, no. 6, pp. 729-736, Jun. 1994.

[28] E. Chiprout and M. S. Nakhla, “Analysis of interconnect networks using complex frequency hopping (CFH),” *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst*., vol. 14, no. 2, pp. 186-200, Feb. 1995.

[29] A. Odabasioglu, M. Celik, and L. T. Pileggi, “PRIMA: passive reduced-order interconnect macromodeling algorithm,” *IEEE Trans. Computer-Aided Design*, vol. 17, no. 8, pp. 645-654, Aug. 1998.

[30] P. Feldmann and R. W. Freund, “Efficient linear circuit analysis by Padé approximation via Lanczos process,” *IEEE Trans. Computer-Aided Design*, vol.14, no. 5, pp. 639-649, May 1995.

[31] M. Silveria, M. Kamon, I. Elfadel, and J. White, “A coordinate-transformed Arnoldi algorithm for generating guaranteed stable reduced-order models of arbitrary RLC circuits”, *Comput. Methods Appl. Mech. Engrg*., vol. 169, no.3-4, pp. 377-389, Feb. 1999.

[32] I. M. Elfadel and D. D. Ling, “A block rational Arnoldi algorithm for multiport passive model order reduction for multiport RLC networks,” *in Proc. IEEE/ACM ICCAD*, 1997, pp. 66-71.

[33] G. Chen and E. G. Friedman, “An RLC interconnect model based on Fourier analysis,” *IEEE Trans. Comput. -Aided Design Integr. Circuits Syst*., vol. 24, no. 2, pp. 170-183, Feb. 2005.

[34] Y. Eo, J. Shim, and W. R. Eisenstadt, “A traveling-wave-based waveform approximation technique for the timing verification of single transmission lines,” *IEEE Trans. Comput. -Aided Design Integr. Circuits Syst*., vol. 21, no. 6, pp. 723-730, Jun. 2002.

[35] J. A. Davis and J. D. Meindl, “Compact distributed RLC interconnect models-Part I: single line transient, time delay and overshoot expression,” *IEEE Trans. Electron Devices*, vol. 47, no. 11, pp. 2068-2077, Nov. 2000.

[36] J. A. Davis and J. D. Meindl, “Compact distributed RLC interconnect models-Part II: Coupled line transient expressions and peak crosstalk in multilevel networks,” *IEEE Trans. Electron Devices*, vol. 47, no. 11, pp. 2078-2087, Nov. 2000.

[37] R. Venkatesan, J. A. Davis, and J. D. Meindl, “Compact distributed RLC interconnect models-Part III: Transients in single and coupled line with capacitive load terminations,” *IEEE Trans. Electron Devices*, vol. 50, no. 4, pp. 1081-1093, Apr. 2003.

[38] R. Venkatesan, J. A. Davis, and J. D. Meindl, “Compact distributed RLC interconnect models-Part IV: Unified models for time delay, crosstalk and repeater insertion,” *IEEE Trans. Electron Devices*, vol. 50, no. 4, pp. 1094-1102, Apr. 2003.

[39] S. Roy, S. and A. Dounavis, “Efficient delay and crosstalk modeling of RLC Interconnects using delay algebraic equations,” *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, vol. 19, no. 2, pp. 342-346, Feb. 2011.

[40] S. Roy and A. Dounavis, “Closed-form delay and crosstalk models for RLC on-chip interconnects using a matrix rational approximation,” *IEEE Trans. Comput. -Aided Design Integr. Circuits Syst*., vol. 28, no. 10, pp. 1481-1492, Oct. 2009.

[41] A.R. Paul, *Analysis of Multiconductor Transmission Lines*, 2nd ed., Wiley Interscience, 2008.

[42] S. Roy and A. Dounavis, “Hybrid approach for accelerated convergence of waveform relaxation based simulation of package/board power distribution networks,” *2013 IEEE MTT-S International Microwave Symposium Digest (MTT)*, 2013, pp. 1-4.

[43] K. Singhal, J. Vlach, and M. Nakhla, “Absolutely stable, high order method for time domain solution of networks,” *Arch. Elek. Übertragung*., vol. 30, pp. 157-166, 1976.

[44] L. Lu, M. S. Nakhla, and Q. J. Zhang "A resetting algorithm for transient analysis of coupled transmission line circuit," *IEEE transaction on microwave theory and techniques*, vol. 42, no 3, pp. 494-500, Mar. 1994.

[45] J. Vlach and K. Singhal, *Computer methods for circuit analysis and design*, 2nd ed., New York: Van Nostrand Reinhold, 1994.

[46] P. Brehonnet, N. Tanguy, P. Vilbe, and L.C. Calvez, "An alternative method for numerical inversion of Laplace transforms," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 53, no. 6, pp. 434-437, Jun. 2006.

[47] P. Gómez, L.Vergara, R. N. Guillén, and F. P. E. Cortés, "Two-dimensional definition of the numerical Laplace transform for fast computation of transient profiles along power transmission lines," *IEEE Trans. Power Del*., vol. 31, no. 1, pp. 412-414, Feb. 2016.

[48] T. R. Arabi, A.T. Murphy, T.K. Sarkar, R.F. Harrington, and A.R. Djordjevic, "On the modeling of conductor and substrate losses in multiconductor, multidielectric transmission line systems," *IEEE Transactions on Microwave Theory and Techniques*, vol.39, no.7, pp.1090-1097, Jul. 1991.

[49] R. Wang and O. Wing, "A circuit model of a system of VLSI interconnects for time response computation," *IEEE Transactions on Microwave Theory and Techniques*, vol. 39, no. 4, Apr. 1991.

[50] Sakurai, S. Kobayashi, and M. Noda, "Simple expressions for interconnection delay, coupling and crosstalk in VLSI's," *in Proc. Int. Symp. Circuits and Systems*, Jun. 1991, pp. 2375-2378.

[51] Y. Tanji and H. Asai, "Closed-form expressions of distributed RLC interconnects for analysis of on-chip inductance effects," *in Proc. IEEE Design Automation Conference*, 2004, pp. 810-813.

[52] S. Shin, Y. Eo, and W. R. Eisenstadt, “Analytic models and algorithms for the efficient signal integrity verification of inductance-effect-prominent multicoupled VLSI circuit interconnects,” *IEEE Trans. VLSI Systems*, vol. 12, no. 4, pp. 395-407, Apr, 2004.

[53] K. D. Boese, A. B. Kahng, and G. Robins, “High-performance routing trees with identified critical sinks,” *in Proc. IEEE/ACM Design Automation Conf*., Jun. 1993, pp. 182-187.

[54] K. D. Boese, A. B. Kahng, B. A. McCoy, and G. Robins, “Rectilinear Steiner trees with minimum Elmore delay,” *in Proc. IEEE/ACM Design Automation Conf*., Jun. 1994, pp. 381-386.

[55] S. S. Sapatnekar, “RC interconnect optimization under the Elmore delay model,” *in Proc. IEEE/ACM Design Automation Conf*., Jun. 1994, pp. 387-391.

[56] R. Achar and M. S. Nakhla, “Simulation of high-speed interconnects,” *Proceedings of the IEEE*, vol. 89, no. 5, May 2001.

[57] A. Deutsch, “Electrical characteristics of interconnections for high-performance systems,” *Proceedings of the IEEE*, vol. 86, no. 2, Feb.1998.

[58] J. Vlach, “Numerical method for transient responses of linear networks with lumped, distributed or mixed parameters,” *Journal of the Franklin Institute*, vol. 288, no. 2, pp. 99-113, Aug. 1969.

[59] J. J. Bourquin, “*Stability of a Class of Lumped-Distributed Systems*,” Ph.D. Thesis, Univ. of Ill., 1967.

[60] F. A. Lindholm and W. W. Happ, “Transient analysis of thin-film structures,” *Journal of Brit. IRE*, vol. 26, no. 5, pp. 421-435, Nov. 1963.

[61] G. Doetsch, “Guide to the applications of Laplace Transforms”, pp. 218, 194, rinceton, N.J., D. Van Nostrand.

[62] O. Perron, “Die Lehre von Kettenbruchen”, Teubner, Stuttgart, 1954.

[63] I. Hajj, K. Singhal and J. Vlach, “Efficient analysis of nonlinear networks by piecewise-linear approximation”, *Proc. Eleventh Ann. Allerton Conj. on Circuits and Systems*, pp. 635-642, Oct., 1973.

[64] S. W. Director and R. A. Rohrer, “Automated network design-the frequency-domain case,” *IEEE Trana. Circuit Theory*, vol. CT-16, no. 3, pp. 330-337, Aug. 1969.

[65] S. W. Director, “LU factorisation in network sensitivity calculation,” *IEEE Trana. Circuit Theory*, vol. CT-18, pp. 184-185, 1971.

[66] S. W. Director and R. A. Rohrer, “The generalized adjoint network and network sensitivities”, *IEEE Trans. Circuit Theory*, vol. CT-16, no. 3, pp. 318-323, Aug. 1969.

[67] A. K. Seth and K. Singhal, “Time-domain sensitivity using the adjoint network”, *Electron. Lett*., pp. 563-565, Sept. 1971.

[68] MATLAB and Statistic Toolbox Release 2014a, MathWorks, Natick, MA, USA, 2014.

[69] HSPICE U-2008.09-RA, Synopsys, Mountain View, CA, USA, 2008.

[70] S. G. Talocia and V. Loggia, “Signal integrity verification of complex high-speed interconnects via waveform relaxation,” *Proc IEEE-EMC*, pp. 328-333, Sep. 2011.

[71] V. Loggia, S. G. Talocia, and H. Hu, “Transient simulation of complex high-speed channels via waveform relaxation,” *IEEE Trans. Compon. Packag. Manufact. Tech*., vol 1, no. 11, Nov. 2011.

Curriculum Vitae

Name: Yu Jiang

Post-secondary Education and Degrees:

Northeast Forestry University
Harbin, Heilongjiang, China
1994-1998 B.A.

Northeast Forestry University
Harbin, Heilongjiang, China
1998-2000 M.A.

Harbin Institute of Technology
Harbin, Heilongjiang, China
2003-2006 Ph.D.

Related Work Experience

Associate Professor
Harbin Engineering University
Harbin, Heilongjiang, China
2006-2013

Assistant Professor
The University of Western Ontario
2015

Teaching Assistant
The University of Western Ontario
2018-2019

Publications