DUBLIN CITY UNIVERSITY
SCHOOL OF ELECTRONIC ENGINEERING

A Final Project report on

Simulation and exploration of THz TRANSMISSION LINES

By Mohammed AL Shuaili

BACHELOR OF ENGINEERING

IN

ELECTRONIC AND COMPUTER ENGINEERING

MAJORING IN

THE INTERNET OF THINGS

Supervised by Dr Marissa Condon

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Dr Marissa Condon, for their invaluable guidance, insightful feedback, and unwavering support throughout this research. Their expertise and encouragement were instrumental in shaping this work. I am also sincerely thankful to my colleagues for their constructive critiques and suggestions, which greatly enhanced the quality of this study. Special thanks go to DCU for providing the necessary resources and infrastructure to conduct this research. Finally, I owe immense gratitude to my family and friends for their patience, motivation, and emotional support during this challenging yet rewarding journey.

Declaration

I declare that this material, which I now submit for assessment, is entirely my own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my work. I understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. I have read and understood the Assignment Regulations set out in the module documentation. I have identified and included the source of all facts, ideas, opinions, and viewpoints of others in the assignment references. Direct quotations from books, journal articles, internet sources, module text, or any other source whatsoever are acknowledged and the source cited in the assignment references. This assignment, or any part of it, has not been previously submitted by me or any other person for assessment on this or any other course of study.

I have read and understood the DCU Academic Integrity and Plagiarism policy at
https://www.dcu.ie/policies/academic-integrity-plagiarism-policy

Name: Mohammed Al Shuaili Date: 30/3/2025

Abstract

This report presents the simulation and analysis of terahertz (THz) transmission lines (TLs), addressing the critical challenge of modelling high-frequency behaviour for next-generation technologies such as 6G networks, wireless data centres, and biomedical imaging. Three numerical methods—Finite-Difference Time-Domain (FDTD), RLC ladder networks, and Asymptotic Waveform Evaluation (AWE)—were evaluated against exact solutions derived via Numerical Inverse Laplace Transform (NILT). The FDTD method achieved high accuracy but incurred significant computational costs, while the RLC ladder approach, optimized with adaptive ode23 solvers, demonstrated superior efficiency. For THz simulation, it’s found through experimental testing that including frequency-dependent resistance is essential, AWE and Vector Fitting (VF/MVF) methods outperformed conventional techniques: AWE achieved low RMSE but required stabilization via first-order rational approximations and complex frequency hopping, whereas VF/MVF delivered exceptional accuracy with 24 poles compared to 32 poles for AWE by iteratively relocating poles in the frequency domain. The study further validated these models with trapezoidal and sinusoidal inputs, showing that recursive convolution methods enable accurate transient analysis at THz frequencies. Key findings emphasize the trade-offs between discretization granularity, computational efficiency, and accuracy, providing actionable insights for optimizing TL models. This work advances THz transmission lines design by establishing validated modelling frameworks, with AWE/VF-MVF emerging as the preferred approach for high-frequency, frequency-dependent applications.

Table of Contents

Table of Figures

Table of Tables

Chapter 1 – Introduction

The growing demand for wireless data rates exceeding 100 Gbit/s, driven by applications such as 5G backhaul/fronthaul, data centres, and intra-device links, has spurred interest in terahertz (THz) frequencies (0.1–10 THz) [1]. This spectrum offers vast bandwidths, enabling high-speed THz communications for 6G networks and applications such as remote sensing, biomedical imaging, and space communication [2]. However, challenges such as developing accurate channel models, reliable RF components, and standardized measurement methods remain [1]. Recent hardware advancements, using both electronic and photonic approaches, demonstrate progress, but further research is essential to bridge the gap between theoretical potential and practical deployment.

This project aims to develop and validate efficient numerical models for THz transmission lines to predict signal behaviour and optimize system performance. The study addresses key propagation challenges at THz frequencies, such as high attenuation and dispersion, which are critical for the design of next-generation communication systems. To achieve this, the project evaluates original methods and techniques proposed from 1995 to recent years, assessing their suitability for THz applications. In addition, the research focuses on developing an accurate model using Y-parameters and Asymptotic Waveform Evaluation (AWE). The developed model is then compared with the original methods to determine their effectiveness and limitations at THz frequencies.

Three primary methods are used in this study: Finite-Difference Time-Domain (FDTD) for transient analysis, Numerical Inverse Laplace Transform (NILT) for obtaining the exact response from the exact s-domain solution, and RLC ladder approximation for computational efficiency. The FDTD method provides a foundation for simulating time-domain electromagnetic wave propagation, while the exact model offers precise frequency-domain insights and the NILT response serving as a benchmark for evaluating other methods. The RLC ladder approximation, on the other hand, is used to simplify complex transmission line models, making them computationally tractable for large-scale simulations. These methods are compared against each other and to the obtained model using Y-parameters and Asymptotic Waveform Evaluation (AWE) to evaluate their effectiveness for THz applications. Factors affecting their performance, such as the number of sections, computational cost, and accuracy, are analysed to determine the most efficient approach for THz transmission line modelling.

The report includes a literature review, implementation details of these modelling methods, simulation results, challenges faced, and future work plans. By integrating Y-parameters and AWE into the modelling framework, this research aims to establish a validated numerical approach for THz transmission line modelling. The proposed model addresses the limitations of earlier methods and offers the best computationally efficient and accurate solution for THz signal analysis. By comparing the developed model with original methods from 1995 to most recent years, this study provides valuable insights into the evolution of THz modelling techniques and their applicability to modern communication systems. This research aims to advance THz communication technologies by designing high-performance systems capable of operating at high frequencies.

1.2 Summary

This project develops and validates numerical models for THz transmission lines to optimize system performance and address propagation challenges. Three primary methods—FDTD, NILT (exact solution), and RLC ladder approximations—are evaluated for accuracy, computational efficiency, and effectiveness compared to the obtained model using AWE. Additionally, MATLAB software package is used for the implementation, but these methods can be applied using any coding language or software.

Chapter 2- Technical Background

Modelling THz (0.1–10 THz) transmission lines require understanding wave propagation, transmission line theory, and numerical methods. At THz frequencies, the behaviour of transmission lines is governed by the Telegrapher's equations (1), which describe the relationship between voltage and current along the line. These equations are derived from Maxwell's equations and are given by [3]:

$$\frac{dv(x,t)}{dx} = -R(x)i(x,t) - L(x)\frac{di(x,t)}{dt} \tag{1a}$$

$$\frac{di(x,t)}{dx} = -G(x)v(x,t) - C(x)\frac{dv(x,t)}{dt} \tag{1b}$$

where $v(x,t)$ and $i(x,t)$ represent the voltage and current at position x and time t, respectively. $R,\ L,\ G,$ and C are the per-unit-length resistance, inductance, conductance, and capacitance of the transmission line. At THz frequencies, these parameters become highly frequency-dependent, making accurate modelling more complex.

2.1 Finite-Difference Time-Domain (FDTD) Method

The FDTD method is a widely used numerical technique for solving electromagnetic problems, particularly in the time domain. It discretizes the transmission line into small segments as in Figure 2.1, allowing for the simulation of voltage and current over time. The FDTD method is based on approximating the derivatives in the Telegrapher's equations using finite differences [4]. In this approach voltages (v_n) are calculated at the ends of each section, while currents (i_n) are computed at the middle of each section as illustrated in Figures 2.1 and 2.2. Then, v_n and i_n can be derived as in equations 2a and 2b.

$$v_k^{n+1} = v_k^n - \frac{\Delta\mathrm{t}}{\Delta\mathrm{xC}}\left(i_k^{n+\frac{1}{2}} - i_{k-1}^{n+\frac{1}{2}}\right) \tag{2a}$$

$$i_{k-1}^{n+3/2} = i_k^{n+1/2} - \frac{\Delta\mathrm{t}}{\Delta\mathrm{xL}}(v_{k+1}^{n+1} - v_k^{n+1}) \tag{2b}$$

This method provides a foundation for simulating transient and steady-state behaviours of transmission lines, but it can be computationally intensive, especially for long lines or high frequencies.

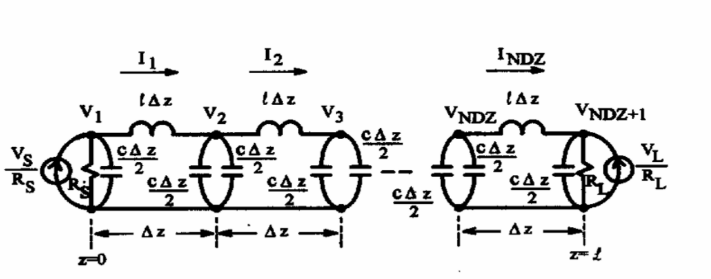

Figure 2.1: Equivalent representation of the transmission line using the Lumped Pi circuit model, illustrating the discretization of the line [4]

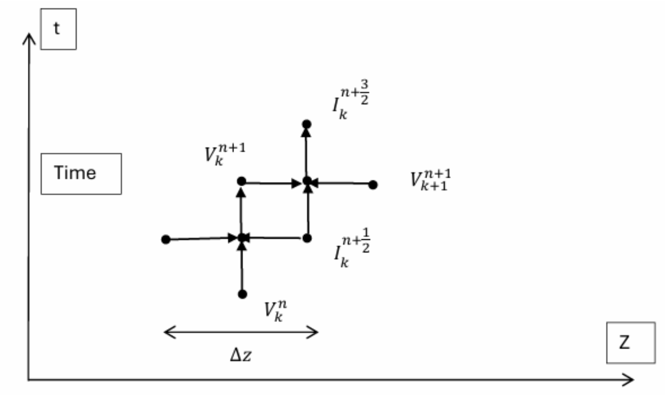

Figure 2.2: Staggered grid representation for the FDTD method, illustrating the spatial (z) and temporal (t) discretization. The voltage (V) is defined at grid points, while the current (I) is defined at the midpoints between the grid points [4]

2.2 Numerical Inverse Laplace Transform (NILT)

The NILT method is an L-stable method for converting frequency-domain solutions into time-domain solutions for simulating transient phenomena in multiconductor transmission line (MTL) systems [6][8]. NILT0, the baseline variant, approximates time-domain responses using residues and poles of a Padé rational function, enabling sparse time points with L-stability which damps errors exponentially for stiff systems, ensuring stability even with large time steps. L-stability as explained in equation (3) [7][11], prevents oscillations and guarantees accurate transient simulations with sparse sampling. NILTn (for *n*≥1) extends this by recursively computing high-order derivatives of Laplace-domain solutions, reducing truncation errors a factor of (n+1) N+M, where *n* represents the derivative order, and *N* and *M* are parameters of the Padé rational approximation that define the degrees of the numerator and denominator, respectively. Importantly, NILTn achieves this improvement while preserving numerical stability [6].

A general first-order differential equation can be expressed as:

$$\frac{dy}{dt} = \gamma y \tag{3a}$$

Applying the backward difference method to equation (4), we obtain:

$$\frac{y_n - y_{n-1}}{h} = \gamma y_n \tag{3b}$$

Rearranging the equation:

$$y_n(1 - h\gamma) = y_{n-1} \tag{3c}$$

$$\frac{y_n}{y_{n-1}} = \frac{1}{1 - h\gamma} \tag{3d}$$

For numerical stability, we consider the following conditions:

- If $\left|\frac{1}{1-h\gamma}\right| < 1$, it is an A-stable method. (3e)
- If $\left(\frac{1}{1-h\gamma}\right)^n \rightarrow 0\ as\ n \rightarrow \infty$ the method is L-stable. (3f)

Furthermore, NILTcv is based on the Bromwich integral which is a mathematical foundation for converting Laplace-domain solutions F(s) to time-domain functions f(t). This integral is numerically evaluated using the Fast Fourier Transform (FFT) and the quotient-difference (q-d) algorithm. The algorithm computes coefficients d_n by iteratively updating quotients and differences of series terms, enabling accurate approximation with fewer terms. Thus, it transforms a slowly converging infinite series into a continued fraction which converges faster and more efficiently. The time-domain function *f*(*t*) is approximated using a discrete form derived from the Laplace transform *F*(*s*) as in equation (4). The approximation involves a finite sum evaluated by the FFT and an infinite sum accelerated by the q-d algorithm, which uses a continued fraction to improve convergence [8][9].

$$f(t) = \frac{1}{2\pi j}\int_{c-\infty}^{c+\infty} F(s)e^{st}ds \tag{4}$$

This approach allows for the exact solution of the transmission line's behaviour in the s-domain, which can then be compared with approximate methods such as the RLC ladder to validate accuracy.

2.3 RLC Ladder Approximations

The RLC ladder method approximates a transmission line by dividing it into multiple sections, each represented by lumped resistive (R), inductive (L), and capacitive (C) elements as shown in Figure 2.3 [3][5]. This discretization simplifies the transmission line into a network of interconnected RLC circuits, making it easier to model and simulate.

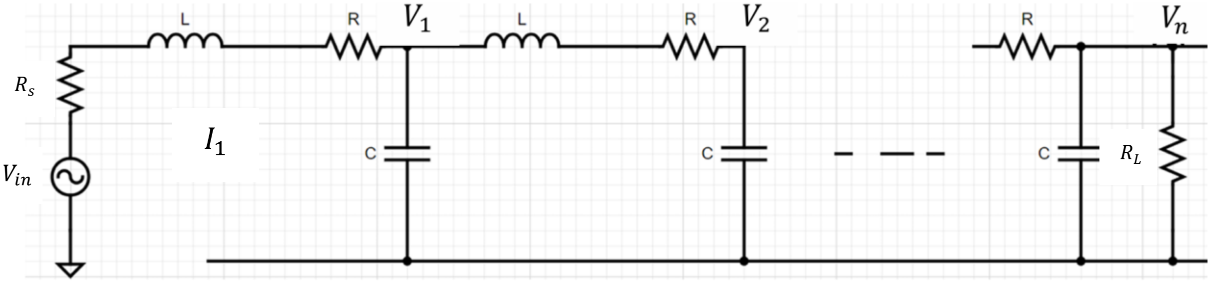

Figure 2.3: RLC ladder network approximates a transmission line with N sections and lumped elements (R), (L), and (C).

Governing equations:

Considering one section of the RLC ladder, the following equations are derived:

$$V_s - V_1 = (R + R_s)I_1 dz + L\frac{dI_1}{dt}dz \tag{5}$$

$$I_1 - I_2 = C\frac{dV_1}{dt}dz \tag{6}$$

here, dz represents the length of a small segment of the transmission line and is defined as:

$$dz = \frac{l}{N} \tag{7}$$

where *l* is the total length of the line, and *N* is the number of sections or RLC circuits used to model the line.

$$\frac{dI_1}{dt} = -\frac{1}{L}V_1 - \frac{R_s + R}{L}I_1 + \frac{1}{L}V_s \tag{8}$$

$$\frac{dV_1}{dt} = \frac{1}{C}I_1 - \frac{1}{C}I_2 \tag{9}$$

$$\frac{dI_n}{dt} = -\frac{1}{L}V_n - \frac{R}{L}I_n + \frac{1}{L}V_{n-1} \tag{10}$$

$$\frac{dV_n}{dt} = \frac{1}{C}I_n - \frac{1}{C}I_{n+1} \tag{11}$$

The impedances *R*, *L*, and *C* are defined per unit length (i.e., per *dz*) in Equations (5) and (6). By rearranging Equations (5) and (6), Equations (8) and (9) are obtained. For the *n*-th section, the generalised forms are given by Equations (10) and (11). The accuracy of the RLC ladder approximation depends on the number of sections used; more sections generally lead to higher accuracy but at the cost of increased computational complexity. However, this lumped element model is not optimal for large-scale or broadband simulations due to its inherent discretization errors.

2.4 Asymptotic Waveform Evaluation (AWE):

Asymptotic Waveform Evaluation (AWE) is a computationally efficient technique used to approximate the transient response of large linear systems, such as electrical interconnects, by reducing their high-order state-space models into lower-order approximations. Traditional methods such as methods in SPICE become prohibitively slow for circuits with hundreds of nodes (e.g., PEEC models), but AWE addresses this by extracting dominant poles and residues from the system's transfer function using Padé approximation and moment matching [10]. By focusing on these critical poles, AWE converts the state-space representation—which describes the system's dynamics through differential equations—into a simplified time-domain model.

2.4.1 A general definition of moments in AWE:

The q^{th} moment is defined as:

$$H(s=0) = \int_0^\infty h(t)dt \tag{12a}$$

$$H^{(1)}(s=0) = -\int_0^\infty h(t)tdt \tag{12b}$$

$$H^{(2)}(s=0) = \int_0^\infty h(t)t^2dt \tag{12c}$$

$$H^{(3)}(s=0) = -\int_0^\infty h(t)t^3dt \tag{12d}$$

Application of moments to represent the transfer function $H(s)$:

$$H(s) = \int_0^\infty h(t)\left(1 - st + \frac{1}{2}s^2t^2 - \frac{1}{6}s^3t^3 + \frac{1}{24}s^4t^4\right)dt \tag{13a}$$

$$= H(0) + sH^{(1)}(0) + \frac{1}{2}s^2H^{(2)}(0) + \frac{1}{6}s^3H^{(3)}(0) + \cdots \tag{13b}$$

$$= m_0 + m_1s + m_2s^2 + m_3s^3 + \cdots \tag{13c}$$

$$= \sum\nolimits_{k=0}^{\infty} \frac{s^k}{k!}H^{(k)}(s=0) = \sum\nolimits_{k=0}^{\infty} m_ks^k \tag{13d}$$

where

$$m_k = \frac{1}{k!}H^{(k)}(s=0) = \frac{(-1)^q}{q!}\int_0^\infty t^q h(t)dt \text{ [10]} \tag{14}$$

2.5 Y-parameters:

In the analysis of transmission lines, the Y-parameters (admittance parameters) are commonly used to characterize the relationship between the currents and voltages at the input and output ports. These parameters are typically determined experimentally and can be approximated using rational functions of the complex frequency variable (s) as explained in equation (15) [11].

$$\begin{bmatrix} I_s \\ -I_R \end{bmatrix} = \begin{bmatrix} Y_{11} & Y_{12} \\ Y_{21} & Y_{22} \end{bmatrix}\begin{bmatrix} V_s \\ V_R \end{bmatrix} \tag{15}$$

Each one is approximated with a rational function as in equation (16):

$$Y_{ij} = \frac{(a_{nij}s^{n-1}+..+a_{0ij})}{s^n+..+b_{0ij}} \tag{16}$$

In the open-circuit condition, where the output current (I_R) is zero, the voltage transfer function ($\frac{V_R}{V_s}$) is derived as a ratio of these rational functions (17b). This transfer function is then converted into a state-space form, to facilitate further analysis. Finally, AWE is applied to the state-space model to obtain a time-domain representation, enabling the study of the system's transient and steady-state behaviour. This approach provides a systematic method for modelling and simulating the response of transmission lines.

$$\frac{V_R}{V_s} = -\frac{Y_{21}}{Y_{22}} = f\left(\frac{(a_{n21}s^{n-1}+..+a_{021})}{s^n+..+b_{021}}, \frac{(a_{n22}s^{n-1}+..+a_{022})}{s^n+..+b_{022}}\right) \tag{17a}$$

$$= -\frac{\frac{(a_{n21}s^{n-1}+..+a_{021})}{s^n+..+b_{021}}}{\frac{(a_{n22}s^{n-1}+..+a_{022})}{s^n+..+b_{022}}}$$

$$= \frac{f_{n-1}s^{n-1}+\ldots+f_0}{s^n+g_{n-1}s^{n-1}+..+g_0} \tag{17b}$$

2.6 Summary

This section summarises methods for modelling transmission lines, governed by Telegrapher's equations. Key approaches include the FDTD method for transient simulations via staggered discretization, Numerical Inverse Laplace Transform (NILT) variants (e.g., NILTcv using FFT) for frequency-to-time-domain conversion, and RLC ladder networks RLC ladder approximations (lumped-element networks). Furthermore, AWE simplifies high-order systems via dominant poles and Pade approximations, while Y-parameters characterize admittance relationships using rational functions. In addition, complex frequency hopping is implicitly integrated via pole-residue analysis and Bromwich contour methods to enhance stability and convergence in transient simulations [8] [11]. These methods address challenges in simulating transient and steady-state behaviours of THz lines, prioritizing trade-offs between accuracy, stability, and computational efficiency.

Chapter 3- Design of Transmission lines models

3.1.1 The exact solution of Transmission lines (Lumped Element Model):

The exact solution of the transmission line can be derived using the π-section model—which represents the precise formulation of the RLCG ladder in the frequency domain—as follows [11][8].

Firstly, write the Telegrapher equations (1) in state space form

$$\frac{d}{dx}\begin{bmatrix} v(x,t) \\ i(x,t) \end{bmatrix} = \begin{bmatrix} 0 & -R(x) \\ -G(x) & 0 \end{bmatrix}\begin{bmatrix} v(x,t) \\ i(x,t) \end{bmatrix} - \begin{bmatrix} 0 & -L(x) \\ -C(x) & 0 \end{bmatrix}\frac{d}{dt}\begin{bmatrix} v(x,t) \\ i(x,t) \end{bmatrix} \qquad (18)$$

moving the representation in (18) to the Laplace domain,

$$\frac{d}{dx}\begin{bmatrix} V(x,s) \\ I(x,s) \end{bmatrix} = \begin{bmatrix} 0 & -Z(x,s) \\ -Y(x,s) & 0 \end{bmatrix}\begin{bmatrix} V(x,s) \\ I(x,s) \end{bmatrix} + \begin{bmatrix} 0 & L(x) \\ C(x) & 0 \end{bmatrix}\begin{bmatrix} V(x,0) \\ I(x,0) \end{bmatrix} \qquad (19)$$

where $Z(x,s) = R(x) + sL(x)$, and $Y(x,s) = G(x) + sC(x)$ are series impedance. Now, let:

$$W(x,s) = \begin{bmatrix} V(x,s) \\ I(x,s) \end{bmatrix}, M = \begin{bmatrix} 0 & -Z(x,s) \\ -Y(x,s) & 0 \end{bmatrix}, and\ N = \begin{bmatrix} 0 & L(x) \\ C(x) & 0 \end{bmatrix}$$

where

$$\frac{dW(x,s)}{dx} = MW(x,s) + NW(x,0) \qquad (20)$$

then, $W(l,s)$ at the end of the transmission line is equal to,

$$W(l,s) = \Phi\, \mathrm{W(0,s)} + \int_0^l e^{M(l-x)} NW(x,0)dx \qquad (21)$$

where

$$\Phi = e^{Ml}, so\ \phi = \begin{bmatrix} \Phi_{11} & \Phi_{12} \\ \Phi_{21} & \Phi_{22} \end{bmatrix} \qquad (22)$$

considering zero initial conditions $(i.e, at W(x,0) = 0)$then,

$$W(l,s) = \Phi\, \mathrm{W(0,s)} \qquad (23a)$$

or

$$W(l,s) = \begin{bmatrix} V(l,s) \\ I(l,s) \end{bmatrix} = \begin{bmatrix} \Phi_{11} & \Phi_{12} \\ \Phi_{21} & \Phi_{22} \end{bmatrix} \begin{bmatrix} V(0,s) \\ I(0,s) \end{bmatrix} \qquad (23b)$$

considering open voltage transmission line (i.e, $I(l,s) = 0$) then,

$$\begin{bmatrix} V(l,s) \\ 0 \end{bmatrix} = \begin{bmatrix} \Phi_{11} & \Phi_{12} \\ \Phi_{21} & \Phi_{22} \end{bmatrix} \begin{bmatrix} V(0,s) \\ I(0,s) \end{bmatrix} \qquad (23c)$$

$$0 = \Phi_{21} V(0,s) + \Phi_{22} I(0,s) \tag{23d}$$

$$I(0,s) = -\Phi_{22}^{-1}\Phi_{21} V(0,s)$$

$$V(l,s) = \Phi_{11} V(0,s) + \Phi_{12}\, I(0,s) \tag{24}$$

$$V(l,s) = \Phi_{11} V(0,s) - \Phi_{12}\Phi_{21}\Phi_{22}^{-1} V(0,s) \tag{25}$$

re arrange equation (25) and simplify, the following is achieved in the s domain,

$$V(l,s) = \frac{2V_s(s)e^{l\sqrt{YZ}}}{e^{(2l\sqrt{YZ})} + 1} \tag{26}$$

where $V_s(s)$ is the input in the s domain. However, if the source resistance is considered then equation (25) can be expressed as equation (27e) and thus, $V(l,s)$ is equal to equation (28a).

$$V(0,s) = V_s - R_s I(0,s) \tag{27a}$$

$$V(0,s) = V_s - R_s(-\Phi_{22}^{-1}\Phi_{21}V_s) \tag{27b}$$

$$V(0,s) = \frac{V_s}{1 - R_s\Phi_{21}\Phi_{22}^{-1}} \tag{27c}$$

$$V(l,s) = \frac{V_s}{1 - R_s\Phi_{21}\Phi_{22}^{-1}}\,(\Phi_{11} - \Phi_{12}\,\Phi_{21}\,\Phi_{22}^{-1}) \tag{27e}$$

$$\frac{V(l,s)}{V_s} = \frac{2e^{\left(l(YZ)^{\frac{1}{2}}\right)}(YZ)^{\frac{1}{2}}}{(YZ)^{\frac{1}{2}} - R_sY + e^{\left(2l(YZ)^{\frac{1}{2}}\right)}(YZ)^{\frac{1}{2}} + R_sYe^{\left(2l(YZ)^{\frac{1}{2}}\right)}} \tag{28a}$$

Now, let,

$$x = \sqrt{YZ\,},$$

$$V(l,s) = \frac{V_s(s)2e^{lx}}{e^{(2lx)} + 1}\ or\ (with\ R_s)\ V(l,s) = \frac{2xe^{lx}V_s}{e^{lx}\big((x + R_sY)e^{-lx} + (x + R_sY)e^{lx}\big)} \tag{28b}$$

using the fact that,

$$cosh(x) = \frac{(e^x + e^{-x})}{2}\ and\ ,\sinh(x) = \frac{-e^x + e^x}{2} \tag{28c}$$

Then,

$$\frac{V(l,s)}{Vs(s)} = \frac{1}{\cos\mathrm{h}\big(l\sqrt{YZ}\big)}\ or\ (with\ R_s) = \frac{\sqrt{YZ}}{\sqrt{YZ}\cosh\big(l\sqrt{YZ}\big) + R_sYsinh\big(l\sqrt{YZ}\big)} \tag{29}$$

One then can use NILTcv as in code 5 in the appendix [9], to simulate this with different values of R, L, C and G, and ultimately compare the output to the approaches mentioned in chapter 2.

3.1.2 Exact solution using (impedance and admittance):

In this approach, the transmission line is represented as a cascade of small segments, each consisting of a series impedance (Z_{series}) and a parallel admittance ($Y_{parallel}$) as illustrated in Figure 3.1. These elements correspond to the physical properties of the transmission line [11].

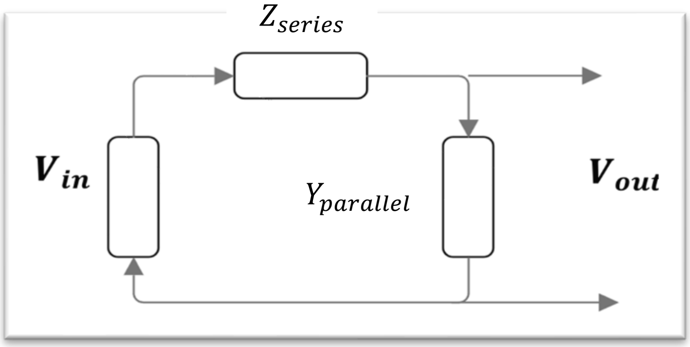

Figure 3.1: Equivalent circuit representation of a transmission line segment with series impedance (Z_series) and parallel admittance (Y_parallel).

where,

$$Z_{series} = Z_o \sinh(\gamma l) \tag{30a}$$

$$Z_o = \sqrt{\frac{Z}{Y}}\,, \qquad Y = G + sC, \qquad Z = (R + sL), \quad \gamma = \sqrt{ZY} \tag{30b}$$

$$Y_{parallel} = Y_o \tanh\left(\frac{\gamma l}{2}\right), \quad Y_o = \frac{1}{Z_o}\,, \quad Z_{parallel} = \frac{1}{Y_{parallel}} \tag{30c}$$

From Figure 3.1, the following equation of the transfer function is obtained:

$$T(s) = \frac{V_o}{V_{in}} = \frac{Z_{parallel}}{Z_{series} + Z_{parallel}} \tag{30d}$$

$$T(s) = \frac{V_o}{V_{in}} = \frac{\dfrac{\sqrt{\dfrac{R+sC}{G+sC}}}{\tanh\left(l\dfrac{\sqrt{(R+sC)(G+sC)}}{2}\right)}}{\sqrt{\dfrac{R+sC}{G+sC}}\sinh\left(l\sqrt{(R+sC)(G+sC)}\,\right) + \dfrac{\sqrt{\dfrac{R+sC}{G+sC}}}{\tanh\left(l\dfrac{\sqrt{(R+sC)(G+sC)}}{2}\right)}} \tag{30e}$$

$$T(s) = \frac{V_o}{V_{in}} = \frac{\dfrac{1}{\tanh\left(l\dfrac{\sqrt{(R+sC)(G+sC)}}{2}\right)}}{\sinh\left(l\sqrt{(R+sC)(G+sC)}\right) + \dfrac{1}{\tanh\left(l\dfrac{\sqrt{(R+sC)(G+sC)}}{2}\right)}} \tag{30f}$$

using the fact that,

$$\tanh\left(\frac{x}{2}\right) = \frac{\cosh(x) - 1}{\sinh(x)}$$

$$T(s) = \frac{V_o}{V_{in}} = \frac{1}{\cosh\left(l\sqrt{(R+sC)(G+sC)}\right)} \tag{31}$$

which is the same as in equation (29).

3.2 AWE implementation:

AWE is a technique used to identify the dominant poles of a system that would otherwise have an extremely large number of poles if modelled precisely. The corresponding residues reflect the significance of these poles in determining a specific response [10]. AWE involves 4 main steps that gives the impulse response in the form of equation (32):

$$h(t) = k_0\delta(t) + k_1e^{p_1t} + \cdots + k_ne^{p_nt} \tag{32}$$

3.2.1 Step 1: Form a state – space representation out of a model (RLC ladder or general TF):

Consider 2 sections of the RLC ladder in Figure 2.3, the following equations are derived:

$$v_{in} = (R_s + R_{dz})i_1 + L_{dz}\frac{di_1}{dt} + v_1 \tag{33a}$$

$$v_1 = R_{dz}i_2 + L_{dz}\frac{di_2}{dt} + v_{out} \tag{33b}$$

$$i_1 - i_2 = C\frac{dv_1}{dt} \tag{33c}$$

$$i_2 = C\frac{dv_o}{dt} \tag{33d}$$

Let,

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} i_1 \\ v_1 \\ i_2 \\ v_o \end{bmatrix} \tag{34a}$$

Rewriting the equations,

$$\frac{di_1}{dt} = -\frac{(R_s + R_{dz})}{L_{dz}} i_1 - \frac{v_1}{L_{dz}} + \frac{v_{in}}{L_{dz}} \tag{34b}$$

$$\frac{di_2}{dt} = \frac{-R_{dz} i_2}{L_{dz}} - \frac{v_{out}}{L_{dz}} + v_1 \tag{34c}$$

$$\frac{dv_1}{dt} = \frac{1}{C}(i_1 - i_2) \tag{34d}$$

$$\frac{dv_o}{dt} = \frac{1}{C}\, i_2 \tag{34e}$$

Then the state space representation is:

$$A = \begin{bmatrix} \frac{-R_s + R_{dz}}{L_{dz}} & -\frac{1}{L_{dz}} & 0 & 0 \\ \frac{1}{C} & 0 & -\frac{1}{C} & 0 \\ 0 & \frac{1}{L_{dz}} & -\frac{R_{dz}}{L_{dz}} & -\frac{1}{L_{dz}} \\ 0 & 0 & \frac{1}{C} & 0 \end{bmatrix}, B = \begin{bmatrix} \frac{1}{L_{dz}} \\ 0 \\ 0 \\ 0 \end{bmatrix}, \qquad C = [0 \quad 0 \quad 0 \quad 1]\ and\ D = 0 \tag{35}$$

In general, the RLC ladder can be expressed in state space form following the same pattern. This is coded in code 4 in the appendix, where it generates matrix A, B and C based on n number of sections.

Generally, the matrix definitions (for n stages):

1. State matrix A (size 2n x 2n)
   - Inductor rows (odd indices 1, 3, ….., 2n-1)

$$A[k,k] = -\frac{R_s + R_{dz}}{L_{dz}}, (first\ \ inductor), A[k,k] = -\frac{R_{dz}}{L_{dz}}\ (subsequent). \tag{36}$$

$$A[k, k+1] = -\frac{1}{L_{dz}}, \qquad A[k, k-1] = \frac{1}{L_{dz}}\ \ (if\ k > 1). \tag{37}$$

   - Capacitor rows (even indices 2, 4, ….., 2n).

$$A[k, k-1] = \frac{1}{C}, A[k, k+1] = -\frac{1}{C} \qquad (if\ k < 2n). \tag{38}$$

2. Input matrix B (size 2n x 1).

$$B = \left[\frac{1}{L_{dz}}, 0, 0, \dots \dots, 0\right]^T. \tag{39}$$

3. Output matrix C (size 1 x 2n):

$$C = [0,0, \dots \dots ,0,1]. \tag{40}$$

Now, considering a general form of a transfer function that is:

$$H(s) = \frac{V_o}{V_i} = \frac{(a_{n-1}s^{n-1} + a_{n-2}s^{n-2} + \cdots + a_0)}{(s^n + b_{n-1}s^{n-1} + b_{n-2}s^{n-2} \dots + b_0)} \tag{41}$$

Equation (41) can then be converted into a state space form as follows [11]:

$$\frac{Z(s)}{V_i} \; and \; \frac{V_o}{Z(s)}$$

$$\begin{tikzcd} \boxed{\frac{1}{s^n + b_{n-1}s^{n-1} + .. + b_0}} \arrow[r, "Z(s)"] & \boxed{a_{n-1}s^{n-1} + a_{n-2}s^{n-2} \dots + a_0} \end{tikzcd} \tag{42a}$$

Then, moving equation (42a) to the time domain gives:

$$\frac{d^n z}{dt^n} + b_{n-1}\frac{d^{n-1}Z}{dt^{n-1}} \dots \dots . + b_0 Z = V_i \tag{42b}$$

and,

$$a_{n-1}\frac{d^{n-1}Z}{dt^{n-1}} + \cdots .. + a_0 Z = V_o \tag{42c}$$

Now, let

$$x_1 = Z \,, x_2 = \frac{dZ}{dt}, x_3 = \frac{d^2Z}{dt^2}, \dots .., x_{n+1} = \frac{d^nZ}{dt^n} \tag{43}$$

That gives the A matrix as:

$$A = \begin{bmatrix} 0 & 1 & \dots . & 0 & 0 \\ 0 & 0 & 1 & .. & 0 \\ .. & .. & .. & .. & 1 \\ -b_0 & -b_1 & -b_2 & \dots .. & -b_{n-1} \end{bmatrix} \tag{44}$$

The B matrix:

$$B = \begin{bmatrix} 0 \\ 0 \\ \dots \\ 1 \end{bmatrix}, (nx1) entries \tag{45}$$

and the C matrix:

$$C = [a_0 \quad a_1 \quad ..... \quad a_{n-1}] \tag{46}$$

One can make the rational fraction in (41) proper (numerator degree < denominator degree) by preforming polynomial division as follows.

$$H(s) = \frac{N(s)}{D(s)} \tag{47a}$$

where,

$$N(s) = Q(s)D(s) + R(s) \tag{47b}$$

Then the proper form is:

$$H(s) = Q(s) + \frac{R(s)}{D(s)} \tag{47c}$$

where $deg(R) < deg(D)$, and the D matrix in the state space equal to $Q(s)$.

3.2.2 Step 2: Compute the moments associated with the system:

Looking at the general form of Y(s):

$$sX(s) = AX(s) + BU(s) \tag{48a}$$

$$Y(s) = C^T X(s) \tag{48b}$$

For impulse input, U(s) = 1.

$$Y(s) = C^T (sI - A)^{-1} B \tag{49a}$$

If equation (49a) is expanded about $s = 0$.

$$Y(s) = C^T(-A)^{-1}B - C^T(-A)^{-2}B(s) + C^T(-A)^{-3}Bs^2 \text{ ...} \tag{49b}$$

However, since:

$$Y(s) = m_0 + m_1 s + m_2 s^2 + .. + m_n s^n \tag{50}$$

Thus,

$$m_0 = -C(A)^{-1}B \tag{51a}$$

$$m_1 = Y'(s_0) = -C^T(A)^{-2}B \tag{51b}$$

$$m_2 = \frac{Y''(s_0)}{2!} = -C^T(A)^{-3}B \tag{51c}$$

$$m_k = \frac{Y^{(k)}(s_0)}{k!} = -C^T(A)^{-(k+1)}B \tag{51d}$$

However, if $s = s_0$, then expanding (49a)

$$Y(s) = C^T(s_0 I - A)^{-1}B - C^T(s_0 I - A)^{-2}B(s - s_0). + \cdots \tag{52a}$$

Thus,

$$m_0 = C(s_0I - A)^{-1}B \tag{52b}$$

$$m_1 = Y'(s_0) = -C^T(s_0I - A)^{-2}B$$

$$m_2 = \frac{Y''(s_0)}{2!} = C^T(s_0I - A)^{-3}B \tag{52c}$$

$$m_k = \frac{Y^{(k)}(s_0)}{k!} = (-1)^k C^T(s_0I - A)^{-(k+1)}B \tag{52d}$$

3.2.3 Step 3: Calculate the poles of the system

In AWE, the poles and residues are determined using the system's moments in order to obtain the time-domain model as outlined in the following equations and derivations [10][9].

Consider a general form of a transfer function.

$$H(s) = \frac{a_0 + a_1s + a_2s^2 + a_3s^3}{1 + b_1s + b_2s^2 + b_3s^3 + b_4s^4} \tag{53}$$

This can be equated to: $H(s) = m_0 + m_1s + m_2s^2 + m_3s^3 + \cdots$ and,

$$\begin{aligned}(1 + b_1s + b_2s^2 + b_3s^3 + b_4s^4)(m_0 + m_1s + m_2s^2 + m_3s^3 + \cdots)\\ = a_0 + a_1s + a_2s^2 + a_3s^3\end{aligned} \tag{54}$$

Multiply and equate powers of s in (54) to obtain,

$$s^0: a_0 = m_0$$

$$s^1: a_1 = m_0b_1 + m_1$$

$$s^2: a_2 = m_0b_2 + m_1b_1 + m_2$$

$$s^3: a_3 = m_0b_3 + m_1b_2 + m_2b_1 + m_3$$

Higher powers of s:

$$s^4: 0 = m_0b_4 + m_1b_3 + m_2b_2 + m_3b_1 + m_4$$

$$s^5: 0 = m_1b_4 + m_2b_3 + m_3b_2 + m_4b_1 + m_5$$

$$s^6: 0 = m_2b_4 + m_3b_3 + m_4b_2 + m_5b_1 + m_6$$

$$s^7: 0 = m_3b_4 + m_4b_3 + m_5b_2 + m_6b_1 + m_7$$

Putting this in matrix form to solve for b_i coefficients in general.

$$\begin{bmatrix} m_0 & m_1 & \ldots. & m_{q-1} \\ m_1 & m_2 & \ldots. & m_q \\ m_3 & m_4 & \ldots & .. \\ .. & \ldots & \ldots. & .. \\ m_{q-1} & \ldots & \ldots & m_{2q-1} \end{bmatrix} \begin{bmatrix} b_q \\ b_{q-1} \\ b_{q-2} \\ .. \\ b_1 \end{bmatrix} = -\begin{bmatrix} m_q \\ m_{q+1} \\ m_{q+2} \\ \ldots \\ m_{2q-1} \end{bmatrix} \tag{55a}$$

Gaussian elimination is applied to determine the coefficients ($b_1, b_2, b_3, .., b_q$) with ($q = 4$) in this case. The poles of the system are found by solving ($B(s) = 0$). That is, solve:

$$b_q s^q + b_{q-1} s^{q-1} \ldots.. + b_1 s + 1 = 0 \tag{55b}$$

which gives the poles of the system.

3.2.4 Step 4: find the residues:

Generalised approach to determining the residues:

$$h(t) = \sum_{j=1}^{q} k_j e^{p_j t} \tag{56a}$$

in the s domain:

$$H(s) = \sum_{j=1}^{q} \frac{k_j}{s - p_j} = \frac{a_0 + a_1 s + a_2 s^2 + \cdots + a_{q-1} s^{q-1}}{1 + b_1 s + b_2 s^2 + b_3 s^3 + \cdots + b_q s^q} \tag{56b}$$

$$\frac{k_j}{s - p_j} = k_j \left(\frac{1}{s - p_j}\right) = -\frac{k_j}{p_j}\left(\frac{1}{1 - \frac{s}{p_j}}\right) \tag{56c}$$

Let: $x = \frac{s}{p_j}$, Thus

$$\left(1 - \frac{s}{p_j}\right)^{-1} = 1 + \left(\frac{s}{p_j}\right) + \left(\frac{s}{p_j}\right)^2 + \left(\frac{s}{p_j}\right)^3 + \cdots \tag{57}$$

Hence,

$$H(s) = \sum_{j=1}^{q} -\frac{k_j}{p_j}\left(1 + \left(\frac{s}{p_j}\right) + \left(\frac{s}{p_j}\right)^2 + \left(\frac{s}{p_j}\right)^3 + \cdots\right) \tag{58}$$

since $H(s)$ is as in equation (50), hence

$$m_0 = -\left(\frac{k_1}{p_1} + \frac{k_2}{p_2} + \cdots + \frac{k_q}{p_q}\right)$$

$$m_1 = -\left(\frac{k_1}{p_1^2} + \frac{k_2}{p_2^2} + \cdots + \frac{k_q}{p_q^2}\right)$$

$$m_{2q-1} = -\left(\frac{k_1}{p_1^{2q}} + \frac{k_2}{p_2^{2q}} + \cdots + \frac{k_q}{p_q^{2q}}\right)$$

which can be solved using $V\Lambda k = -m$ where

$$V = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \frac{1}{p_1} & \frac{1}{p_2} & \cdots & \frac{1}{p_q} \\ \frac{1}{p_1^2} & \frac{1}{p_2^2} & .. & \frac{1}{p_q^2} \\ . & . & . & . \\ \frac{1}{p_1^{q-1}} & \frac{1}{p_2^{q-1}} & . & \frac{1}{p_q^{q-1}} \end{bmatrix}, \Lambda = \begin{bmatrix} \frac{1}{p_1} & 0 & .. & .. & 0 \\ 0 & \frac{1}{p_2} & 0 & .. & 0 \\ 0 & 0 & . & . & 0 \\ 0 & 0 & .. & \frac{1}{p_{q-1}} & 0 \\ 0 & 0 & 0 & .. & \frac{1}{p_q} \end{bmatrix}, k = \begin{bmatrix} k_1 \\ k_2 \\ . \\ . \\ k_q \end{bmatrix}, m = \begin{bmatrix} m_0 \\ m_1 \\ \cdot \\ \cdot \\ m_{q-1} \end{bmatrix} \tag{59}$$

Thus $k = -\Lambda^{-1}V^{-1}m$ which are the residues of the system. Finaly, the poles obtained in (55b) are combined with the residues in (48) to obtain the time domain impulse response in equation (32).

3.3 Adapting AWE to Obtain a Response to a Unit Step Input (integration method):

To avoid computationally expensive explicit convolution, recursive convolution is applied based on the pole-residue representation of the transfer function [12]. For a system described in the Laplace domain as:

$$Y(s) = \frac{k_i}{s + p_i}X(s) \dots\dots.\frac{d}{dt}y(t) + p_i y(t) = k_i x(t) \tag{60a}$$

Multiply both side by $e^{p_i t}$,

$$\frac{e^{p_i t}dy(t)}{dt} + \ e^{p_i t}p_i y(t) = k_i e^{p_i t}x(t) \tag{60b}$$

$$\frac{d}{dt}\left(\ e^{p_i t}y(t)\right) = k_i e^{p_i t}x(t)$$

$$e^{p_i t_n}y(t_n) - e^{p_i t_{n-1}}y(t_{n-1}) = \int_{t_{n-1}}^{t_n} k_i e^{p\tau}x(\tau)d\tau \tag{60c}$$

Assume $x(\tau)$ is piecewise constant x_n over each time interval $[t_n - t_{n-1}]$. Solving equation (60a) over a time interval using the recursive solution is:

$$y(t_n) = k_\infty x(t_n) + \sum_{i=1}^{q} y_i'(t_n) \tag{61}$$

where

$$y_i'(t_n) = \int_{t_{n-1}}^{t_n} k_i e^{p\tau} x(\tau) d\tau \,/ e^{p_i t_n} + e^{p_i(t_n - t_{n-1})} y_i'(t_{n-1}) \tag{62}$$

The equation in (62) updates the output at t_n using only the previous state $y_i'(t_{n-1})$ and the input $x(t_{n-1})$ eliminating the need to store the entire history of $x(t)$ [10]. Thus, for a unit step response (62) is equal to:

$$y_i'(t_n) = \frac{k_i\left(1 - e^{p_i(t_n - t_{n-1})}\right)}{-p_i} + e^{p_i(t_n - t_{n-1})} y_i'(t_{n-1}) \tag{63}$$

3.4 Implementation of Y parameters:

Considering the general form derived in equation (17) with $n = 2$. One can generate rational expression from a given Y values of a transmission line as follows:

$$Y_R + jY_i = \frac{a_1 s + a_0}{s^2 + b_1 s + b_0} \tag{64}$$

At $s = jw$

$$\begin{aligned} &(Y_R + jY_i)(-w^2 + b_1 jw + b_0) = \; a_1 jw + a_0 \\ &(-w^2 Y_R + b_1 jwY_R + b_0 Y_R - w^2 jY_i - b_1 wY_i + jb_0 Y_i) = \; a_1 jw + a_0 \end{aligned} \tag{65}$$

Re-write (65) as:

$$(b_1 jwY_R + b_0 Y_R - b_1 wY_i + jb_0 Y_i) - a_1 jw - a_0 = w^2 Y_R + jw^2 Y_i \tag{66}$$

To find the coefficients the following is applied:

$$A = \begin{bmatrix} -1 & Y_R^1 & 0 & -Y_i^1 w \\ 0 & Y_i^1 & -jw^1 & jwY_R^1 \\ \dots & .. & .. & .. \\ -1 & Y_R^N & 0 & -Y_i^N w \\ 0 & Y_i^N & -jw^N & -w^2 Y_i^N \end{bmatrix}, B = \begin{bmatrix} a_0 \\ b_0 \\ a_1 \\ b_1 \end{bmatrix} \; and \; C = \begin{bmatrix} Y_R^1 w^2 \\ jY_i^1 w^2 \\ \dots \\ Y_R^N w^2 \\ jY_i^N w^2 \end{bmatrix} \tag{67}$$

$where\ A\,B = C\ and\ B = A^{-1}B.$

Once the rational fraction is derived, AWE is used to obtain the response of the transmission line and compare it to the exact results. Moreover, the approach to obtain an efficient model using AWE and Y measurements is as outlined as follows:

3.4.1 Development of a time-domain model from frequency-domain measurements using iterative curve-fitting:

The proposed methodology involves an iterative algorithm to construct a rational approximation model using frequency-domain measurements (100 points). The process is outlined as follows:

1. Initial Model Generation: A rational approximation model is generated using the first frequency points at which the frequency is small, and the oscillation is small (e.g., indices 1–15) via the system identification method described by Equations (42) and (66). This initial model captures the low-frequency behaviour of the system.

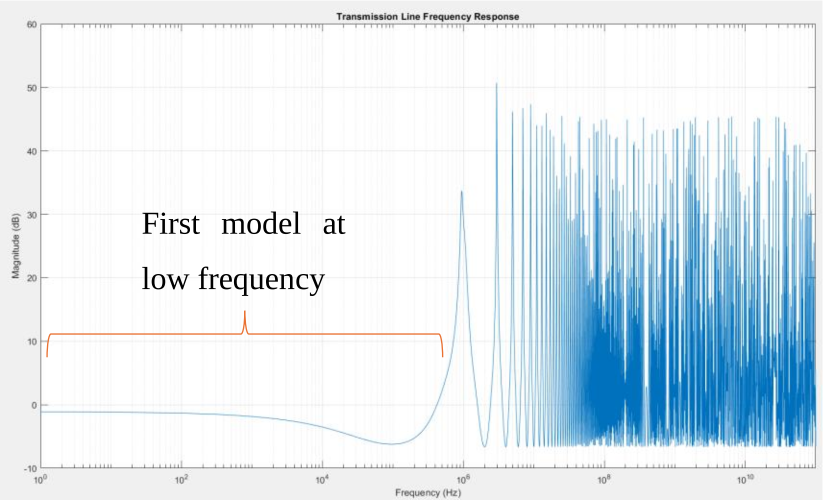

Figure 3.2: The red curve highlights the frequency range where the first model is generated at which the oscillation is minimum.

2. state space model generation: this model is then converted to state space form using equations (42 to 46) then AWE is applied using the first frequency point as its expansion point (e.g., $w_o = 0$) to obtain the impulse response at this range with all associated poles and residues as outlined below:

Let H_o be the first model approximated in step 1 where:

$$H_o = \frac{(a_1 s + a_0)}{s^2 + b_1 s + b_0}, \tag{68}$$

next, using equations (42 to 46) a state space model is formed governing H_o:

$$H_o = \{A_o,\ B_o,\ C_o,\ D_o\}, \tag{69}$$

then, AWE is evaluated around w_o as shown in Figure 3.3:

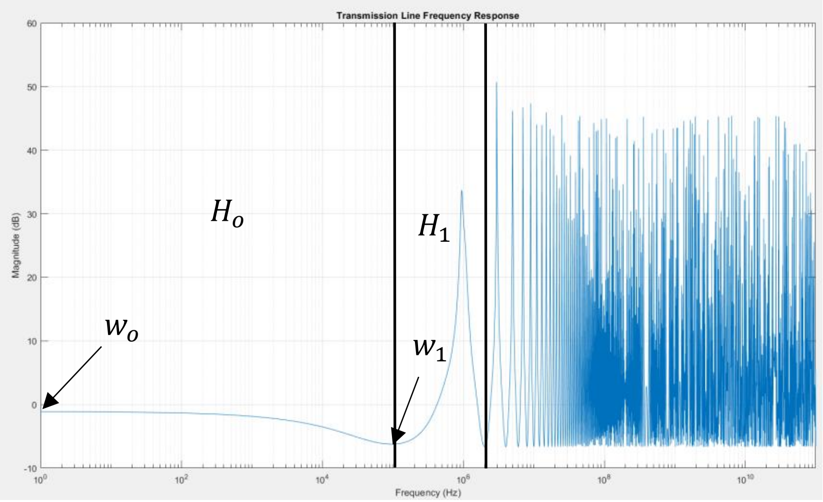

Figure 3.3: presents the expansion points used for AWE evaluation in each model.

The resultant AWE model in the form of equation (70) is then used for the subsequent steps.

$$H_{AWEo} = \frac{r_1}{s - p_1} + \cdots + \frac{r_i}{s - p_i} \tag{70}$$

3. Residual Calculation: The initial model is evaluated at the subsequent desired points (e.g., five frequency points). The difference (residual) between the measured data and the model's prediction at these points is computed to quantify the approximation error as follows:

Let $V_o$be the exact frequency-domain measurements, R_o is the residual and f_{15} is the last frequency point used for the initial model then evaluate the first AWE model at the subsequent frequency points as in Figure 3.3 so that,

$$R_o = V_o(f_{15} .., f_{20}) - H_{AWEo}(f_{15} .., f_{20}) \tag{71}$$

3. Iterative Model Refinement: A secondary rational approximation is generated from the residual error. This correction term is combined with the initial model to enhance accuracy as explained in equation (75). The process repeats iteratively: for each subsequent block of five frequency points (e.g., 15–20, 20–25, etc.), the current composite model is evaluated, the residual is calculated, and a new correction model is generated and superimposed. This step continues for until all desired frequency points are covered.

Let H_1 the subsequent rational approximation using R_o:

$$H_1\{R_o\} = \frac{(a_1 s + a_0)}{s^2 + b_1 s + b_0}, \tag{72}$$

then, a state space representation is created governing H_1, and AWE is evaluated around w_1 as illustrated in Figure 3.2 to obtain the second AWE model:

$$H_1 = \{A_1,\ B_1,\ C_1,\ D_1\}, \tag{73}$$

$$H_{AWE1}\{A_1,\ B_1,\ C_1,\ D_1, w_1\} = \frac{r_1}{s-p_1} + \cdots + \frac{r_i}{s-p_i} \tag{74}$$

Then, the two AWE models are added to enhance accuracy and, the new combined model is used to determine a revised residual error, following the approach outlined in Step 2. The time-domain responses are also computed in every round as described in equation (63) and updated accordingly. In general, the following is applied to achieve the final model:

$$H_{AWE} = H_{AWE1} + \sum_{i=1}^{n} H_{AWEi}\{A_i,\ B_i,\ C_i,\ D_i, w_i\}, \tag{75}$$

$$y(t) = \sum_{i=0}^{n} y_i(t) \tag{76}$$

where H_{AWEi} is generated using the residual error as explained in Steps 2 and 3, and n is the number of desired models used to account for the frequency range.

4. Validation Against Exact Solution: After iterating through all designated frequency blocks, the final composite model's frequency response is compared to the exact solution (precomputed at 100 frequency points) to validate accuracy.

Finally, the final model is pruned by applying post-processing techniques to eliminate redundant and unstable poles and residues, thereby improving numerical stability and computational efficiency. These methods are detailed in subsequent sections.

3.5 Complex frequency hopping:

Complex Frequency Hopping (CFH) is an advanced iterative technique for modelling interconnect networks by combining multi-point moment-matching and pole-residue extraction across the complex frequency plane [14]. CFH enhances accuracy over single-point Pade approximations by strategically "hopping" to expansion points near dominant poles, generating localized Taylor series moments, and synthesizing a unified transfer function. This method efficiently captures high-frequency effects in distributed systems (e.g., transmission lines) while avoiding numerical instability and truncation errors inherent in traditional eigenmode-based moment generation [14].

3.5.1 Multi-Point Moment-Matching and Pole Selection:

CFH generates localized Taylor series moments at strategically chosen expansion points in the complex s*s*-plane (e.g., s=0, s=$j\omega_{max}$, or midpoints between these). At each hop, moments are calculated using equation (77). additionally, poles extracted at each hop are validated through overlapping accuracy regions. If poles from two adjacent hops agree within a tolerance ϵ_{err}, they are marked as accurate. The radius of accuracy R_{acc} around each hop defines the region where poles are accurate (Fig. 3.4) [14]. Spurious or unstable poles are discarded.

$$Y(s)X(s) = E \rightarrow X(s) = \sum_{n=0}^{\infty} M_n(s-p)^n, \tag{77}$$

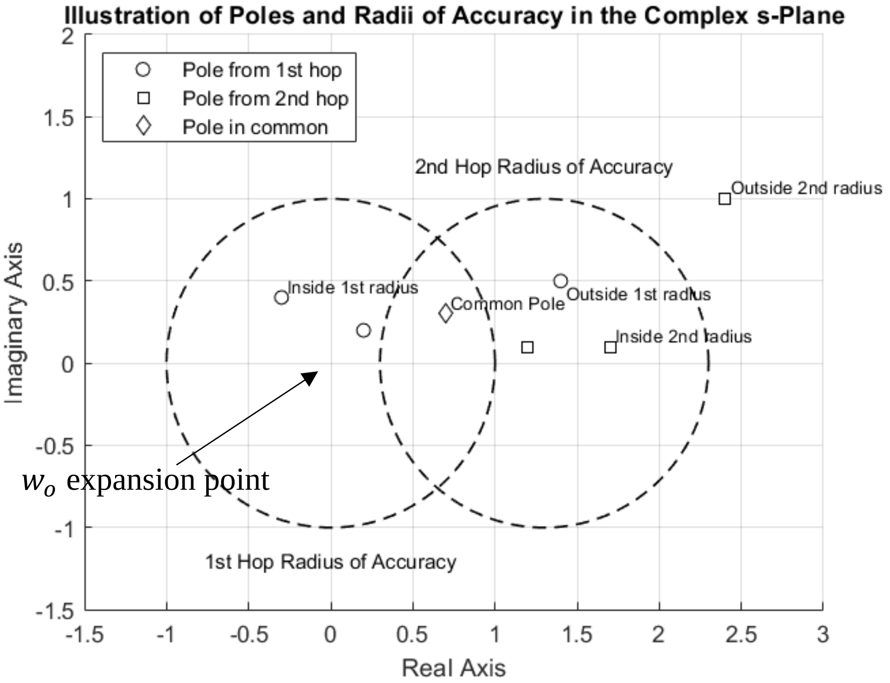

Figure 3.4: Illustration of pole placement and accuracy regions in the complex s-plane [14].

3.5.2 Binary Search Strategy and Unified Transfer Function:

To minimize computational cost, CFH uses a binary search algorithm along the imaginary axis (constrained to the left-half plane for stability). Starting at s=0 and s=$j\omega_{max}$, midpoints are iteratively added until all dominant poles within the frequency range of interest ($\leq f_{max}$) are captured. This ensures coverage of resonant peaks and high-frequency effects. Furthermore, Dominant poles/residues from all hops are combined into a single transfer function (equation 78):

$$H(s) = \sum_{i=1}^{q} \frac{k_i}{s - p_i}, \tag{78}$$

where p_i are stable poles (negative real parts) and k_i are residues. This model avoids numerical instability by rejecting redundant or right-half plane poles [14].

Although CFH is a promising technique for high-frequency interconnect modelling, experimental implementation and testing revealed limitations in extending CFH to terahertz (THz) models, requiring further investigation. The broad spectral span at THz frequencies results in widely separated expansion points, which reduces the likelihood of overlapping poles between adjacent hops. Furthermore, the rapid signal transitions, sensitivity to parasitic effects, and increased computational demands lead to increased errors and numerical rounding issues, especially at high frequencies. These challenges highlight the need for further research into adaptive moment-generation algorithms, enhanced pole-selection criteria, and numerical stability optimizations to tailor CFH for next-generation THz interconnect analysis.

3.6 Development of a model for transmission lines that exhibit fractional-order frequency-dependent behaviour:

Traditional transmission line (T-line) modelling relies on integer-order RLGC (resistance, inductance, conductance, capacitance) parameters, which assume linear frequency dependence for impedance and admittance. However, at terahertz (THz) frequencies (0.1–10 THz), it's observed through experimental testing that these models fail to account for dispersion loss and nonquiet-static effects caused by rapid signal transitions and parasitic coupling. For instance, the skin effect—where current density decreases exponentially with depth into a conductor—introduces frequency-dependent resistance R, while dielectric polarization losses contribute to frequency-dependent conductance G [5][17]. These phenomena are modelled as:

$$R = R_o + R_s(1 + j)\sqrt{w} \tag{79}$$

$$G = G_o + G_D w \tag{80}$$

where R_s and G_D are skin effect and dielectric loss coefficients. Figure (3.5) shows the frequency response with R as a constant and with R as a variable depending on w.

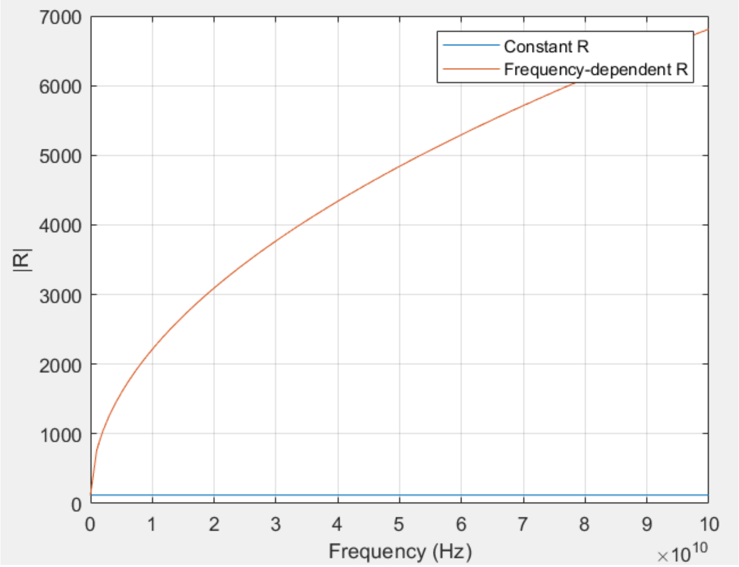

Figure 3.5: Frequency-dependent resistance (R) compared to constant R across a broad frequency range (0–100 GHz).

3.6.1 Transition from *ω* to *s* in NILT-Based Modelling:

To compare TL models with the exact solution, the frequency-dependent skin effect is first incorporated into the model, after which the Numerical Inversion of the Laplace Transform (NILT) is applied. The skin effect, which dominates resistance (*R*) at THz frequencies, is inherently frequency-dependent and is traditionally modelled in the frequency domain as in equation (79). However, to analyse transient responses and ensure causality, the Laplace domain (*s*=*σ*+*jω*) is preferred for its ability to unify steady-state and transient behaviours. This transformation from ω to s is derived as follows:

$$\sqrt{s} \approx \sqrt{jw} = e^{\frac{j\pi}{4}}\sqrt{w}, where\ j = e^{\frac{j\pi}{2}} \tag{81}$$

$$\sqrt{s} = \frac{1+j}{\sqrt{2}}\sqrt{w} \tag{82}$$

$$R = R_o + R_s\sqrt{s}\,\sqrt{2} \tag{83}$$

Thus, the characteristic impedance Z_o in equation (30b) is now:

$$Z_o = \frac{\left(R_o + R_s\sqrt{s}\,\sqrt{2} + sL\right)}{G_o + G_D s\frac{\sqrt{2}}{1+j} + sC}. \tag{84}$$

The remainder of the exact model derivation can be completed by substituting the constant integer parameter with its frequency-dependent counterpart. Subsequently, the numerical inverse Laplace transform (NILT) can be applied to obtain the time-domain representation of the model.

3.7 Summary

This section covers a detailed examination of transmission line (TL) modelling techniques, beginning with the exact solution derived from the lumped element model, where Telegrapher's equations are expressed in state-space form in the Laplace domain, leading to hyperbolic transfer functions (e.g., $(cosh(\frac{l}{\sqrt{ZY}}))$) that account for source resistance effects. The Asymptotic Waveform Evaluation (AWE) method is introduced as a numerical approach to approximate dominant poles and residues, involving state-space formulation of RLC ladders, moment generation via matrix inversion, and recursive convolution for efficient time-domain step response computation. For frequency-domain measurements, an iterative curve-fitting algorithm refines rational approximations by combining residual-corrected models, while Complex Frequency Hopping (CFH) enhances accuracy through multi-point moment-matching and pole selection in the complex s-plane, though challenges persist at terahertz (THz) frequencies due to numerical instability and sparse pole overlap. Additionally, fractional-order frequency-dependent behaviour is addressed by incorporating skin effect and dielectric losses, requiring Laplace-domain transformations and Numerical Inverse Laplace Transform (NILT) for transient analysis. The section bridges theoretical and computational methods, offering scalable solutions for TL modelling across frequency ranges while highlighting limitations in high-frequency regimes.

Chapter 4 – Testing, Results and Discussion

This chapter evaluates the accuracy and efficiency of FDTD, RLC ladder networks and AWE with rational expression models in transmission line analysis, comparing them to exact solutions. Using root mean squared error (RMSE) as in equation (85) and computational CPU speed as metrics (tic toc on MATLAB), the chapter highlights trade-offs between precision and resource demands. Challenges in transient response, AWE's unit step performance, and enhancements via fractional-order model are explored. The chapter concludes with a comparison and guiding method selection for high-speed circuit design based on accuracy and computational cost.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - y_i')^2}\,, or\ \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left|y_i - y_i'\right|^2}\ \ for\ complex\ values\ [11]. \tag{85}$$

4.1 FDTD

Code 1 from the appendix is used with the same parameters for all methods and a load of 100 ohms. The following table and figure highlight the main results.

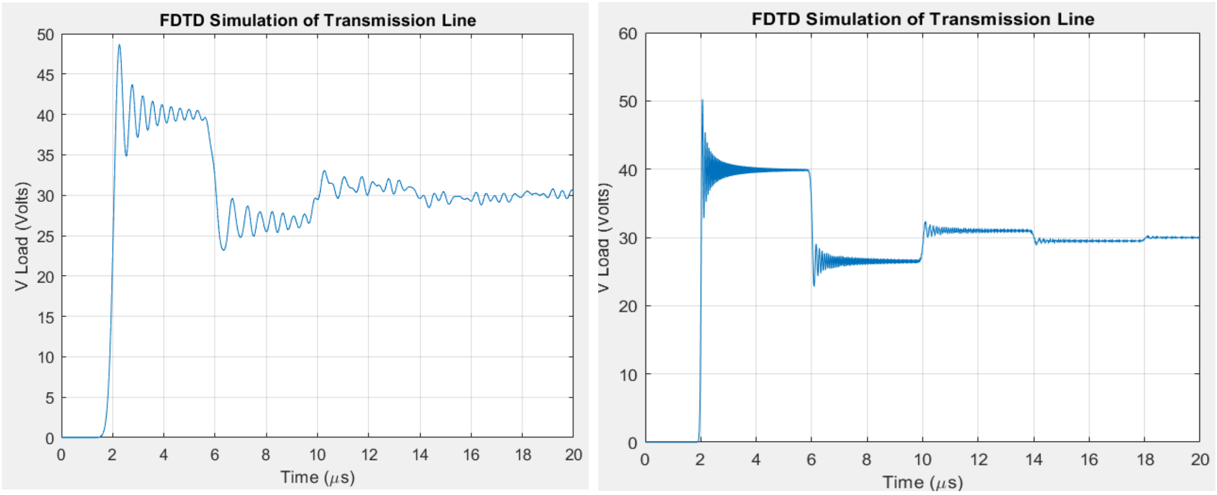

Figure 4.1: FDTD simulation of a transmission line with a 100 Ω load. The left plot corresponds to a discretization of 20 sections, while the right plot uses 200 sections.

Table 1: Comparison of the FDTD method and time taken across different numbers of sections.

Number of sections	Time taken
20	0.0851
100	0.1220
200	0.3180

The Finite-Difference Time-Domain (FDTD) method demonstrates a clear trade-off between accuracy and computational efficiency. As the number of sections increases, the time required for computation grows significantly. For instance, at 200 sections, the time taken is approximately 2.86 seconds, which is much higher compared to the RLC ladder network. The requirement of a very small-time step to maintain accuracy further adds to its computational burden, making it less efficient for large-scale simulations.

4.2 RLC ladder

Code 2 from the appendix is used with the same parameters as in FDTD. The following table and figure show the key results.

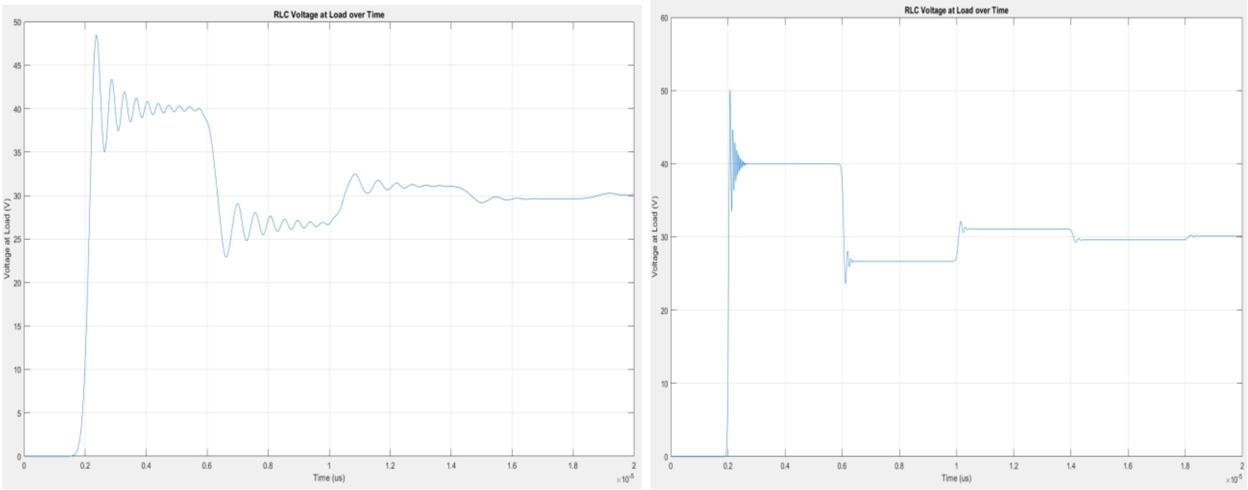

Figure 4.2: Voltage response of an RLC circuit over time using different discretization levels. The left plot corresponds to 20 sections, while the right plot uses 200 sections.

Table 2: Comparison of the RLC method and time taken across different numbers of sections.

Number of sections	Time taken
20	0.0224
100	0.0325
200	0.0515

It was observed that the RLC ladder network, particularly when solved using ode23, was characterized by significantly better computational efficiency compared to FDTD. It was found that even as the number of sections increased, the time required remained relatively low, with 0.0224 seconds recorded for 20 sections, 0.0325 seconds for 100 sections, and 0.0515 seconds for

200 sections. This efficiency was attributed to the variable time-step approach of ode23, which was observed to adapt dynamically to the system's behaviour and reduce unnecessary computations. This is because ode23 uses a 2nd/3rd-order Runge-Kutta pair to estimate and control local error, dynamically adjusting step sizes to optimize efficiency [15] [16]. This approach is particularly effective for non-stiff problems with varying dynamics. Below is an explanation of its variable time-step mechanism [15]:

Let y_n denote the solution at time t_n and compute intermediate stages as follows:

$$k_1 = f(t_n, y_n), \tag{86}$$

$$k_2 = f\left(t_n + \frac{h}{2}, y_n + \frac{h}{2}k_1\right), \tag{87}$$

$$k_3 = f\left(t_n + \frac{3h}{4}, y_n + \frac{3h}{4}k_2\right), \tag{88}$$

$$y_{n+1}^{(3rd)} = y_n + h\left(\frac{2}{9}k_1 + \frac{3}{9}k_2 + \frac{4}{9}k_3\right) \text{ (3rd} - \text{order update)} \tag{89}$$

A 2nd-order solution y_{n+1}^{2nd} is then computed using an additional stage k_4 as in equation (90),

$$k_4 = f\left(t_n + h, y_{n+1}^{(3rd)}\right), \tag{90}$$

$$y_{n+1}^{(2nd)} = y_n + h\left(\frac{7}{24}k_1 + \frac{4}{24}k_2 + \frac{13}{24}k_3\right), \tag{91}$$

The local truncation error e is estimated as the difference between these solutions:

$$e = y_{n+1}^{(3rd)} - y_{n+1}^{(2nd)}. \tag{92}$$

This error drives the adaptive step-size control. The solver enforces a user-defined tolerance tol (or the default one assigned by MTALAB) by scaling the step size h based on the error’s norm ‖e‖. The new step size h_{new} is computed as:

$$h_{\text{new}} = \sigma \cdot h_{\text{old}} \cdot \left(\frac{\text{tol}}{|e|}\right)^{\frac{1}{(p+1)}}, \tag{93}$$

where p=2 (the lower order of the RK pair) and σ≈0.8–0.9 is a safety factor to prevent aggressive step-size changes. If ‖e‖>toll, the step is rejected and repeated with a smaller h; otherwise, h may increase for the next step. The dynamic time-stepping arises because ‖e‖ depends on the solution’s local curvature. In regions where the ODE evolves rapidly (e.g., f(t,y) has large derivatives), ‖e‖ grows, forcing smaller h to maintain accuracy. Conversely, in smoother regions, ‖e‖ decreases, allowing larger h. Furthermore, it was demonstrated that the use of ode23 for transmission line analysis yielded results that were both more accurate and faster compared to those obtained with ode45. It was also shown through fixed time-step analysis that the RLC method performed better in terms of both accuracy and speed when compared to FDTD.

4.3 Exact solution compared with FDTD and RLC

4.3.1 Exact solution:

The exact transmission line model for an open-circuit design is derived using equation (26) or (29). By assuming a lossless line and applying the same parameters as those used in the RLC and FDTD methods, the following results are obtained using code 6 in the appendix [9]. These results are then compared against other models to evaluate their accuracy and computational performance.

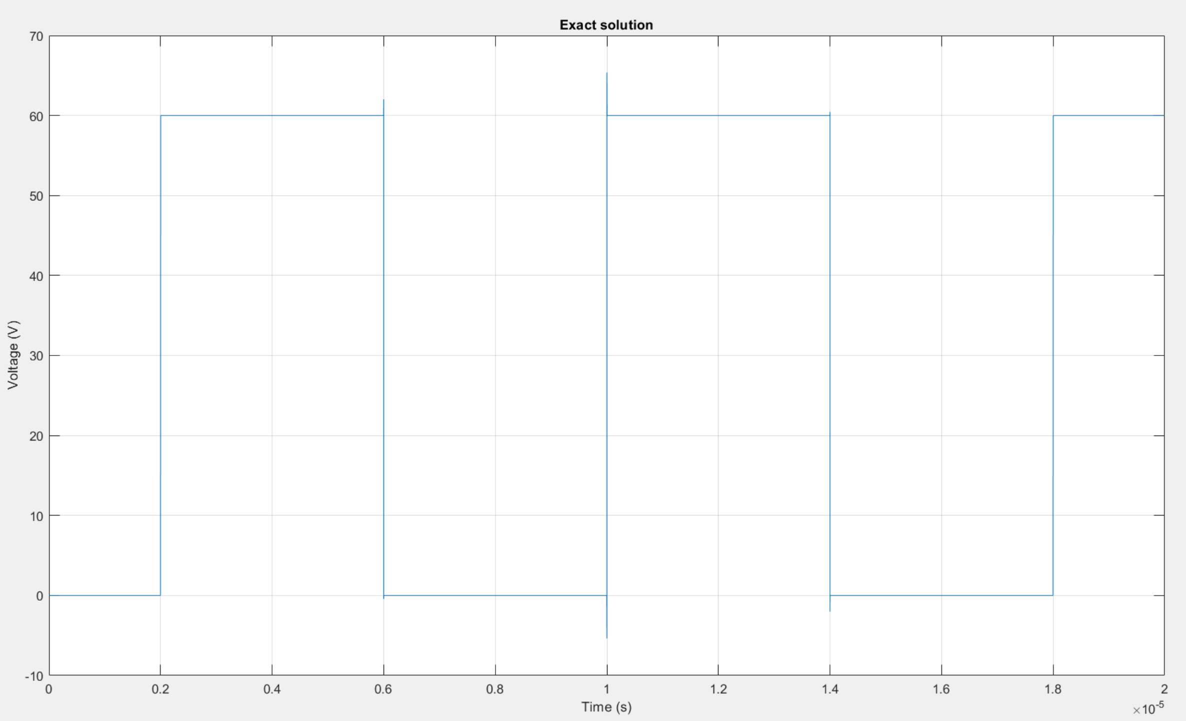

Figure 4.3: Simulation of the exact solution of a lossless transmission line with an open-circuit termination and a 30V input.

The result in Figure 4.3 is expected since the input voltage is 30V, and the transmission line is open-circuited, meaning there is no load. As a result, the wave propagates forward and then reflects backward, leading to a superposition of the incident and reflected waves. This interaction creates a square-like waveform with a peak voltage of 60V, given by $V = V^+ + V^-$.

4.3.2 RLC and FDTD

A fixed time step of $1x10^{-8}$ and $1x10^{-10}$ seconds is used for the RLC and FDTD methods respectively. The table below compares the efficiency of the Finite-Difference Time-Domain

(FDTD) method and the RLC ladder network in terms of Root Mean Square Error (RMSE) and computation time.

Table 3: Comparison of the RLC ladder network and FDTD method across different numbers of sections.

Number of sections	RLC ladder		FDTD	
	Error (RMSE)	time	Error (RMSE)	time
50	9.5367	0.0195	7.2430	0.1142
100	6.9623	0.0395	5.3862	0.1819
200	5.0097	0.0961	5.0746	0.3025
400	3.5005	0.2088	3.2992	0.5047
600	2.8838	0.3722	2.8989	0.8201

It’s observed that when a fixed time step was used for the RLC method, it consistently outperformed FDTD in terms of computational speed while maintaining comparable accuracy. For example, at 400 sections, the RLC method required 0.2088 seconds, compared to 0.5047 seconds for FDTD. The RMSE values indicated that FDTD achieved slightly better accuracy in certain cases (e.g., 5.0746 for 200 sections versus 5.0097 for RLC), but the considerable difference in computation time rendered the RLC method the more practical choice. In addition, it was demonstrated that the RLC method, which does not rely on a small-time step, can be scaled efficiently for higher accuracy.

4.3.3 Visual inspections of all solutions (methods)

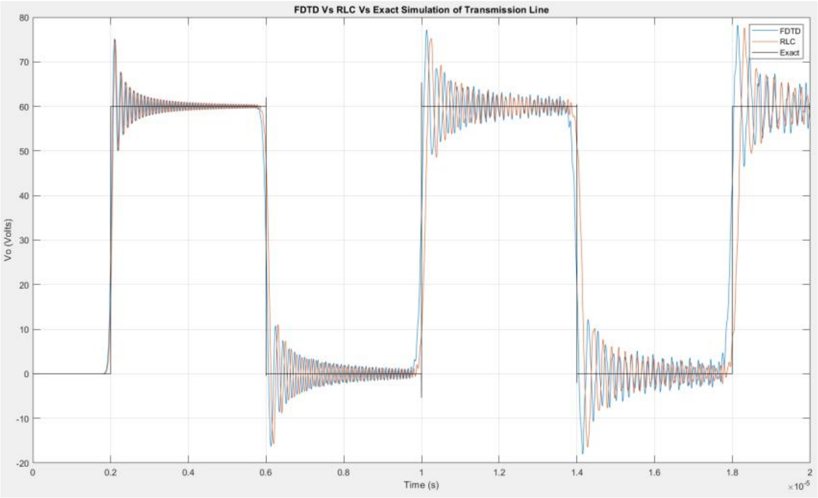

Figure 4.4: Comparison of FDTD, RLC ladder, and exact simulation of a transmission line, showing output voltage Vo over time with numerical oscillations.

4.4 THz Results:

This test evaluates the model's response to three distinct input types: a unit step input, a sine wave with a THz frequency, and a trapezoidal pulse input. The trapezoidal pulse has a rise time of 1 ps from 0 to 1 V, a fall time of 1 ps, and remains at 1 V for 5 ps. The results are based on the following lumped transmission line (TL) parameters.

$$C = 1x10^{-10}F, \qquad L = 250x10^{-9}H, \qquad R = 1200Ohms, \qquad l = 150x10^{-6}m, \qquad R_s = 10 \quad (94)$$

4.4.1 FDTD modification:

In the FDTD method, the equations derived in (2a) and (2b) correspond to the lossless case. To account for a lossy transmission line with per-unit-length parameters and the source resistance R_s , the following modified equations are derived and applied based on [4].

Voltage for k = 1 (at the source) is:

$$V_1^{n+1} = \frac{(R_s \frac{C}{2}\frac{\Delta z}{\Delta t} - \frac{1}{2})V_1^n - \ R_s \left(I_1^{n+\frac{1}{2}}\right) + \frac{(V_s^{n+1} + V_s^n)}{2}}{R_s \frac{C}{2}\ \frac{\Delta z}{\Delta t} + \frac{1}{2}} \qquad (95a)$$

For k = 2, ..,NDZ.

$$V_k^{n+1} = V_k^n - \frac{\Delta t}{\Delta z C}\left(I_k^{n+\frac{1}{2}} - I_{k-1}^{n+\frac{1}{2}}\right) - \frac{G\Delta t}{C} V_k^n. \qquad (95b)$$

if $G\ = 0$, then equation (95b) is identical to the equation for the lossless case, as given in equation (2a).

At K = NDZ+1 the voltage depends on R_L, and should therefore be the same as in the lossless case:

$$V_{NDZ+1}^{n+1} = \frac{(R_L \frac{C}{2}\frac{\Delta z}{\Delta t} - \frac{1}{2})V_{NDZ+1}^{n} - \ R_L\left(I_{NDZ}^{n+\frac{1}{2}}\right) + \frac{(V_L^{n+1} + V_L^{n})}{2}}{R_L \frac{C}{2}\frac{\Delta z}{\Delta t} + \frac{1}{2}} \tag{95c}$$

Finaly, the current update for all nodes (K=1,…,NDZ):

$$I_k^{n+\frac{3}{2}} = I_k^{n+\frac{1}{2}} - \frac{\Delta t}{\Delta zl}(V_{k+1}^{n+1} - V_k^{n+1}) - \frac{R\Delta t}{l} I_k^{n+\frac{1}{2}} \tag{96}$$

These modifications to accommodate the lossy case are implemented in code 7 in the appendix and are consequently applied in generating the following results.

4.4.2- The unit step response:

In this section, the impact of incorporating a source resistance R_s is investigated, highlighting its influence on both computational efficiency and signal accuracy. The inclusion of R_s introduces additional damping into the system, thereby altering the energy dissipation and the effective impedance, which in turn affects the overall performance and error metrics.

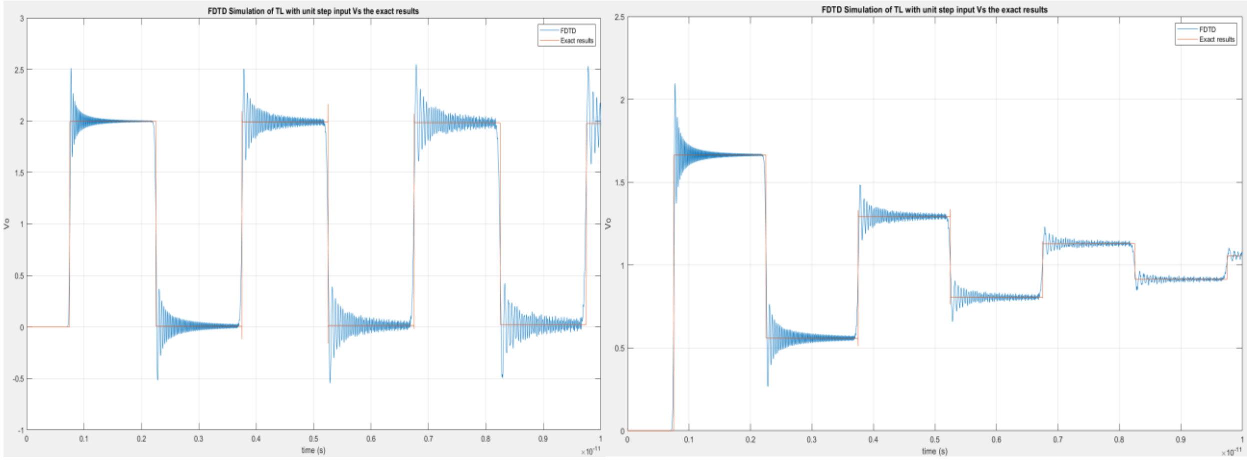

Figure 4.5: Comparison of FDTD simulation results with exact solutions for a transmission line with unit step input. The left plot represents the case without source resistance Rs, while the right plot includes Rs.

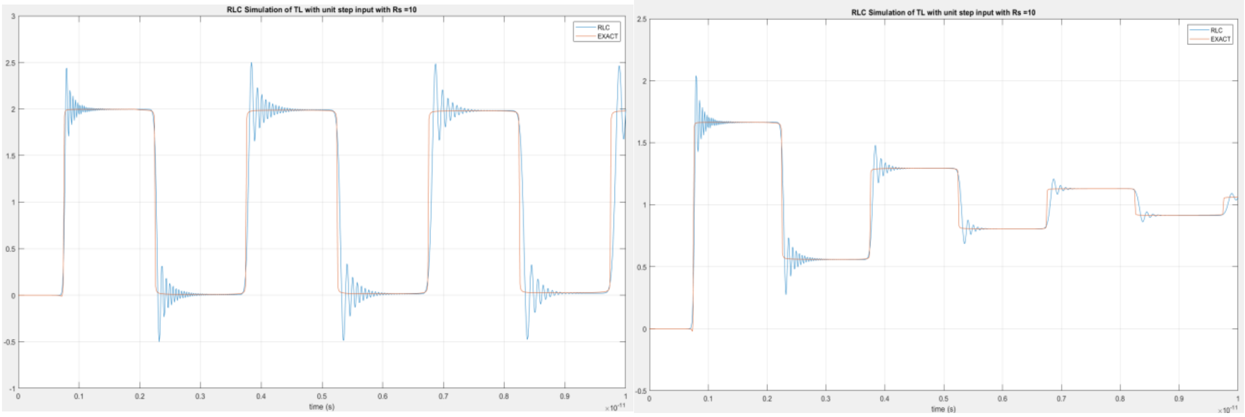

Figure 4.6: Comparison of RLC simulation results with exact solutions for a transmission line with a unit step input. The left plot represents the case without source resistance Rs, while the right plot includes Rs.

Table 4: Comparison of the RLC ladder network and FDTD method across different numbers of sections.

Number of sections	RLC			FDTD		
	Error	Error (Rs)	CPU time	Error	Error (Rs)	CPU time
50	0.3609	0.0998	0.01472	0.2905	0.0836	0.02410
100	0.2516	0.0701	0.02402	0.2060	0.0658	0.04670
200	0.1718	0.0481	0.04665	0.1535	0.0524	0.09126
400	0.1180	0.0333	0.12205	0.1188	0.0421	0.19728

The changes observed when Rs is added are attributed to the additional damping effect it introduces. This damping modifies the transmission line's characteristic impedance and reduces the amplitude of transient oscillations. Consequently, the unit step response is altered, leading to variations in the RMSE errors, as the system dynamics adjust to the resistive influence.

4.4.3- Response to a sine wave input at THz frequency:

This section presents the results of using different model if the input is a sine wave at 100 GHz frequency.

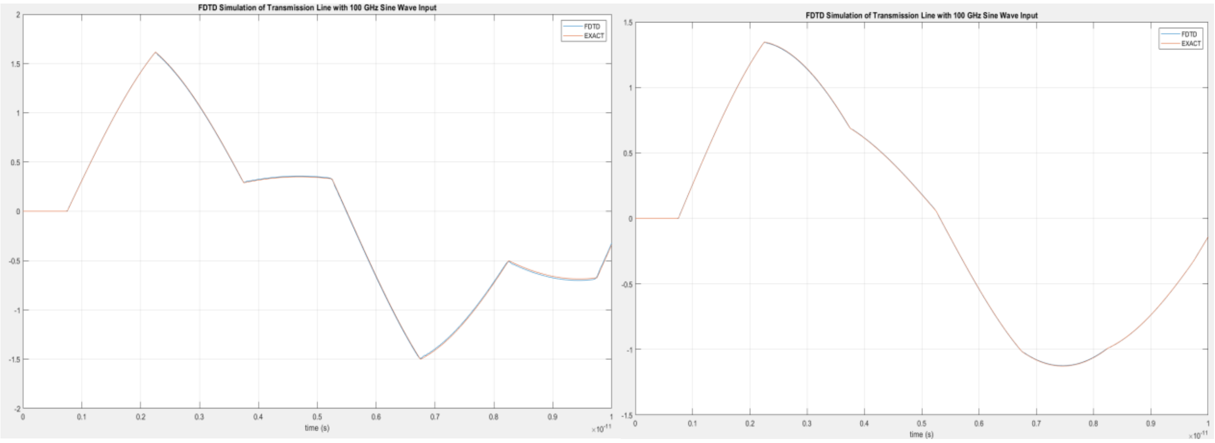

Figure 4.7: Comparison of FDTD simulation results with exact solutions for a transmission line with a sine wave at THz input. The left plot represents the case without source resistance Rs, while the right plot includes Rs.

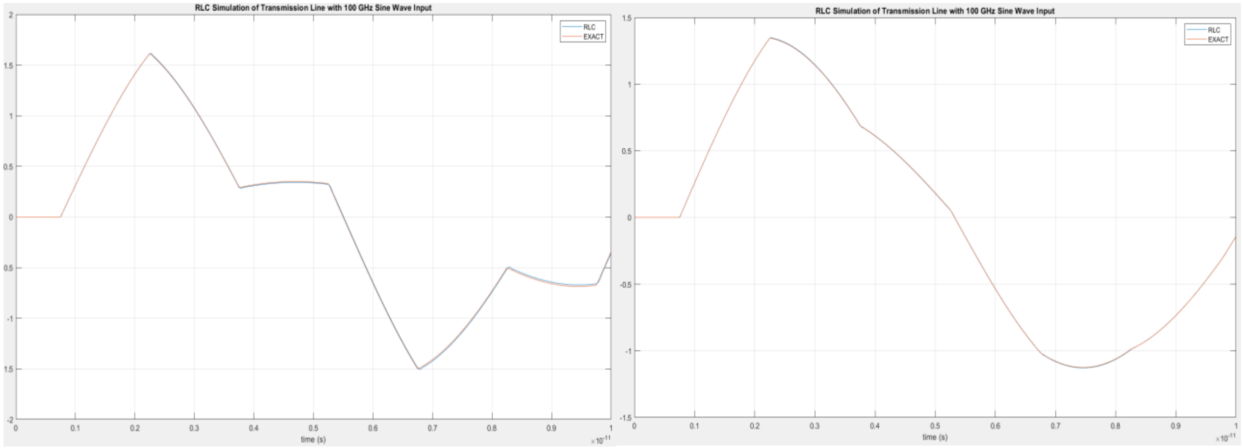

Figure 4.8: Comparison of RLC simulation results with exact solutions for a transmission line with a sine wave at THz input. The left plot shows the case without source resistance Rs, while the right plot includes Rs.

Table 5: Comparison of the RLC ladder network and FDTD method across different numbers of sections with sine wave at THz frequency input.

Number of sections	RLC			FDTD		
	Error	Error (Rs)	CPU time	Error	Error (Rs)	CPU time
50	0.0383	0.0079	0.00716	0.0372	0.0076	0.02410
100	0.0193	0.0039	0.01243	0.0188	0.0038	0.04670
200	0.0097	0.0020	0.02890	0.0095	0.0019	0.09126
400	0.0049	9.8518e-04	0.07670	0.0047	9.6461e-04	0.19728

The inclusion of R_s introduces damping effects, altering the system's transient behaviour and error metrics, as reflected in Table 5. This table demonstrates that both RLC ladder networks and FDTD methods achieve lower errors (e.g., 0.0047 for FDTD at 400 sections) as the number of sections increases, though FDTD requires more CPU time. The analysis underscores the trade-offs between accuracy and computational efficiency when modelling high-frequency signals under resistive influences.

4.4.4- Response due to Trapezoidal pulse:

A trapezoidal pulse that:

- Rises linearly from 0 V to 1 V over 1 ps,
- Stays at 1 V for 5 ps,
- Falls back to 0 V over 1 ps,
  can be built from ramps and exponentials in the Laplace domain.

Time-Domain Definition

$$v_{pulse}(t) = \begin{Bmatrix} 0, & t < 0 \\ linear\ rise, & 0 \le t < 1ps \\ 1, & 1ps \le t < 6ps \\ linear\ fall, & 6ps \le t < 7ps \\ 0, & t \ge 7ps. \end{Bmatrix} \tag{97}$$

Total duration: 7 ps = 1 ps (rise) + 5 ps (high) + 1 ps (fall).

constructing equation (97) via Ramp Functions as follows:

$$ramp(t) = t\,u(t)\,, in\ laplace\ domin \frac{1}{s^2}. \tag{98}$$

A linear rise and fall from 0 to 1 over $[0,\ T_r]$ can be written as

$$\frac{1}{T_r}[ramp(t) - ramp(t - T_r)] \tag{99}$$

Hence, for rise time T_r and peak duration T_P

$$v_{pulse}(t) = \frac{1}{T_r}[ramp(t) - ramp(t - T_r)] - \frac{1}{T_r}\left[ramp\left(t - \left(T_r + T_p\right)\right) - ramp\left(t - \left(2T_r + T_p\right)\right)\right] \tag{100}$$

using $\mathcal{L}\{ramp(t-a)\} = \frac{e^{-as}}{s^2}$, we get:

$$V_{pulse}(s) = \frac{1}{T_r s^2}[1 - e^{-T_r s}] - \frac{1}{T_r s^2}\left[e^{(-T_r + T_p)s} - e^{(-2T_r + T_p)s}\right]. \tag{101}$$

Then $V_{pulse}(s)$ in equation (101) is used for obtaining the exact solution using NILT while $v_{pulse}(t)$ (equation (100) and code 8 in the appendix) is used for RLC and FDTD methods to obtain the following results.

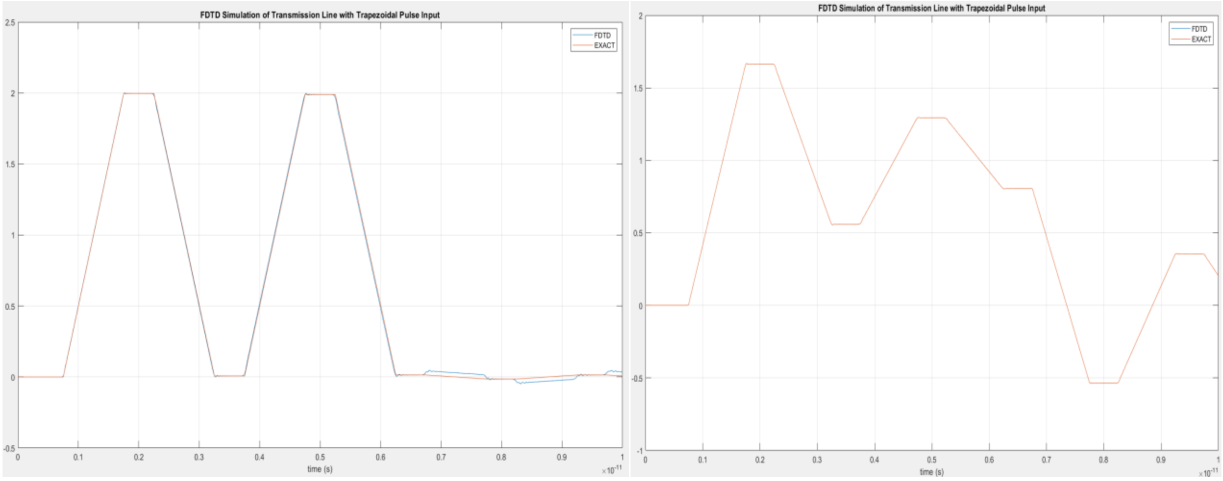

Figure 4.9: Comparison of FDTD simulation results with exact solutions for a transmission line with a pulse input. The left plot represents the case without source resistance Rs, while the right plot includes Rs.

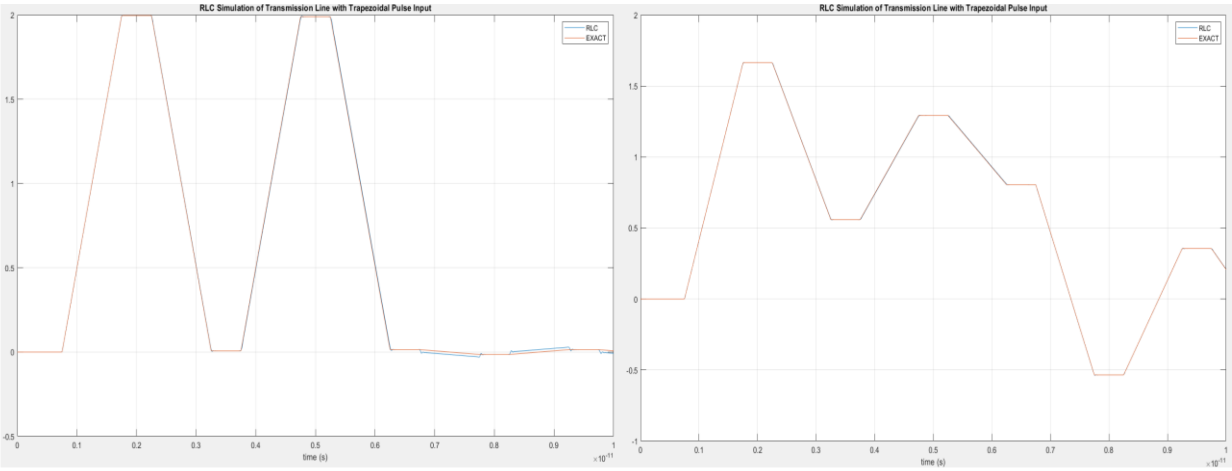

Figure 4.10: Comparison of RLC simulation results with exact solutions for a transmission line with a pulse input. The left plot represents the case without source resistance Rs, while the right plot includes Rs.

Table 6: Comparison of the RLC ladder network and FDTD method across different numbers of sections with trapezoidal pulse input.

Number of sections	RLC			FDTD		
	Error	Error (Rs)	CPU time	Error	Error (Rs)	CPU time

50	0.0700	0.0157	0.00716	0.0690	0.0151	0.14985
100	0.0353	0.0078	0.01243	0.0347	0.0076	0.14901
200	0.0177	0.0039	0.02890	0.0174	0.0038	0.22378
400	0.0089	0.0020	0.07670	0.0087	0.0019	0.32909

This section analyses the response of transmission line models to a trapezoidal pulse input, defined with a 1 ps linear rise to 1 V, a 5 ps plateau, and a 1 ps linear fall. The pulse is mathematically constructed in the Laplace domain using ramp functions (Equations 70–73), combining exponential terms to represent its linear transitions. Simulations using RLC ladder networks and FDTD methods are compared in Table 6, which shows decreasing errors (e.g., RLC error drops from 0.0700 to 0.0089 as sections increase from 50 to 400) but rising computational costs, particularly for FDTD (CPU time increases from 0.14985 to 0.32909). Figures 4.9 and 4.10 illustrate the impact of source resistance R_s, where its inclusion introduces damping effects, aligning simulations closer to exact solutions. The analysis highlights the trade-off between model accuracy and computational efficiency, emphasizing how resistive components refine transient behavior in high-speed pulse applications.

The initial results presented in this section assume a constant resistance *R*, which simplifies the model but neglects the frequency-dependent behaviour critical at THz regimes. This means these results might not be accurate in reflecting the actual behaviour of TL at THz because as discussed in Section 3.6, the skin effect dominates at THz frequencies, causing *R* to scale with ω (or *s* in the Laplace domain). To address this, we integrate the frequency-dependent resistance model derived in Section 3.6.1 and thus obtain the results.

4.4.5 Unit step response (with frequency-dependent resistance)

At terahertz (THz) frequencies, the resistance R becomes highly frequency-dependent as discussed in Section (3.6) impacting the transmission line (TL) response. Using the resistance model given in equation **(**79**)** and the TL parameters outlined in **(**94**)**, one can derive the system's transfer function in the Laplace domain and then, the unit step response is obtained using (NILTcv) Figure (4.11), allowing accurate time-domain analysis while accounting for the high-frequency behaviour of R(f).

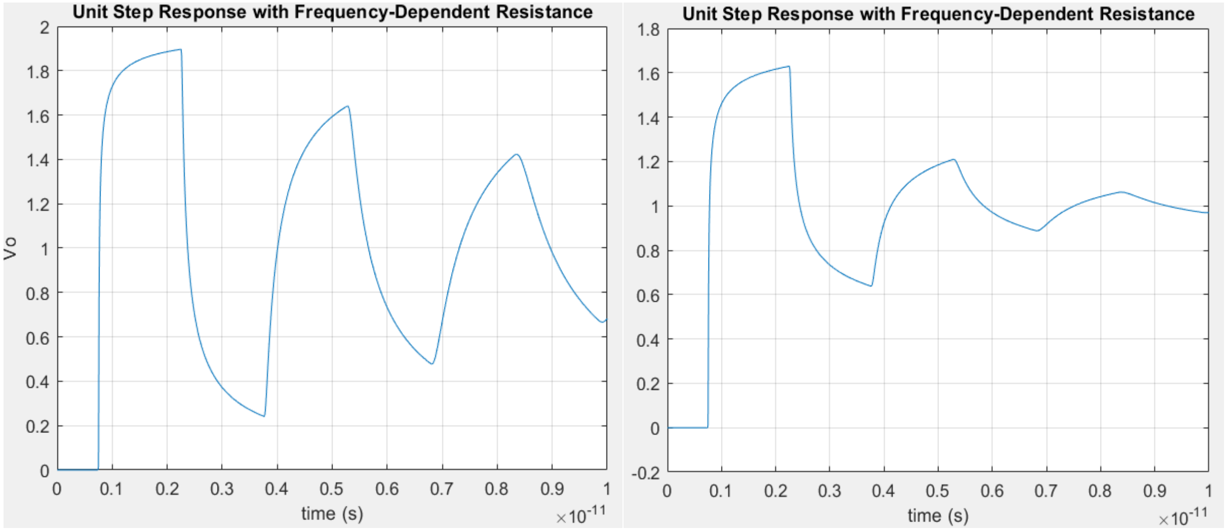

Figure 4.11: The unit step response of the exact model with R being frequency dependent and Rs is zero in the left and 10 in the right.

4.4.6 Response to Sinusoidal and Trapezoidal Inputs (with frequency-dependent resistance)

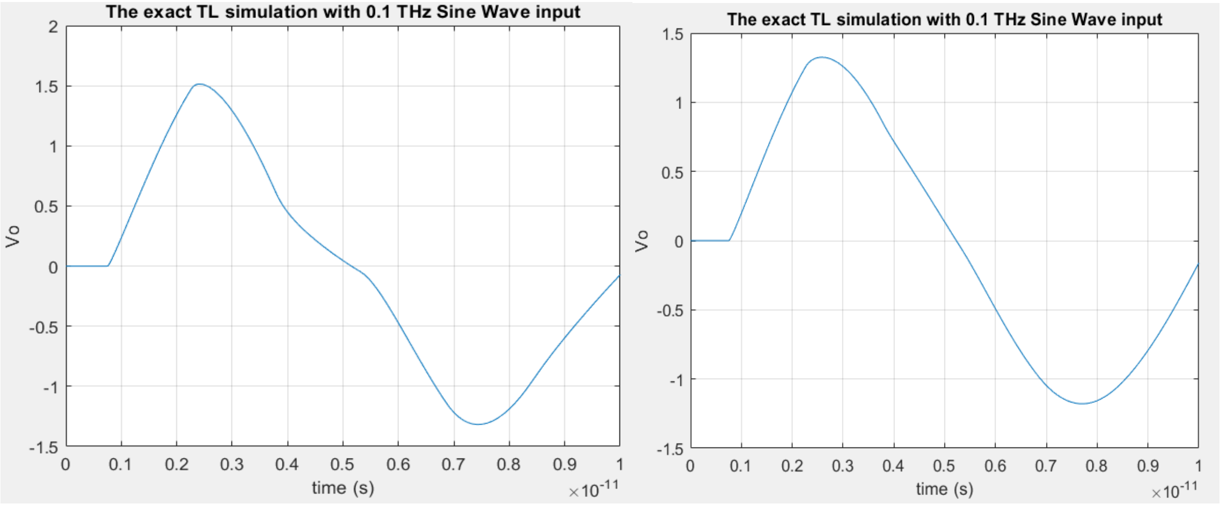

Figure 4.12: The response to sine wave of the exact model with R being frequency dependent and Rs is zero in the left and 10 in the right.

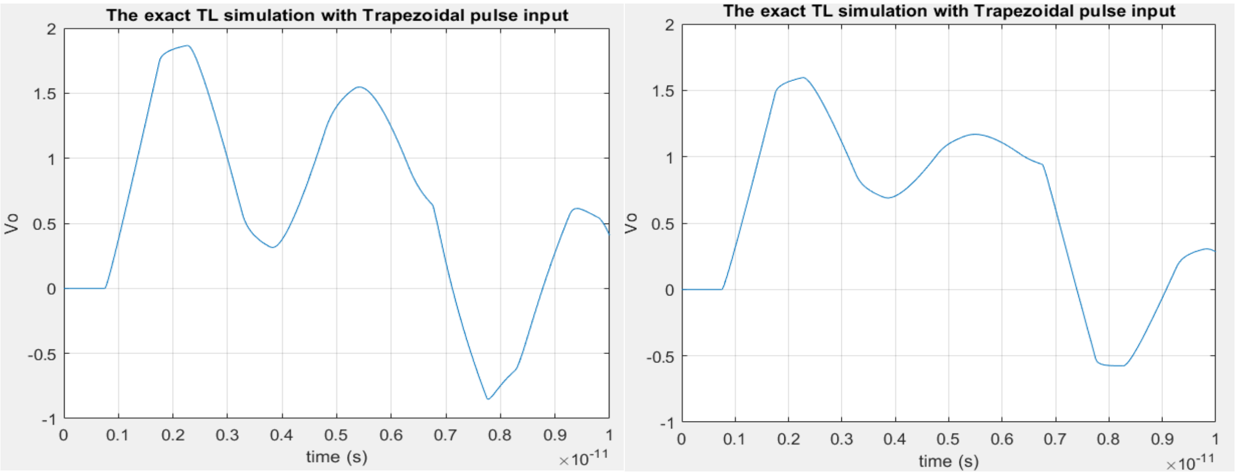

Figure 4.13: The response to trapezoidal pulse of the exact model with R being frequency dependent and Rs is zero in the left and 10 in the right.

Figure (4.12) compares the response of the exact frequency-dependent model to sinusoidal inputs under two conditions: Rs=0 (left) and Rs=10 (right). The inclusion of Rs introduces damping, moderating oscillations and improving agreement with theoretical expectations. Similar behaviour is observed for trapezoidal inputs (Figure 4.13), where frequency-dependent R mitigates waveform distortion might be caused by dispersion or skin effects.

4.4.7 Challenges in Integrating Frequency-Dependent Resistance

Incorporating frequency-dependent R into time-domain methods such as RLC and FDTD presents significant challenges, as these methods inherently assume time-invariant parameters. Initial attempts to address this include:

1. Rational Approximation of R(w): A pole-residue model (Figure 4.14) was derived from the frequency response of R(w), and its impulse response was computed via equation (32). However, integrating this time-domain approximation into RLC and FDTD models yielded discrepancies compared to exact solutions.

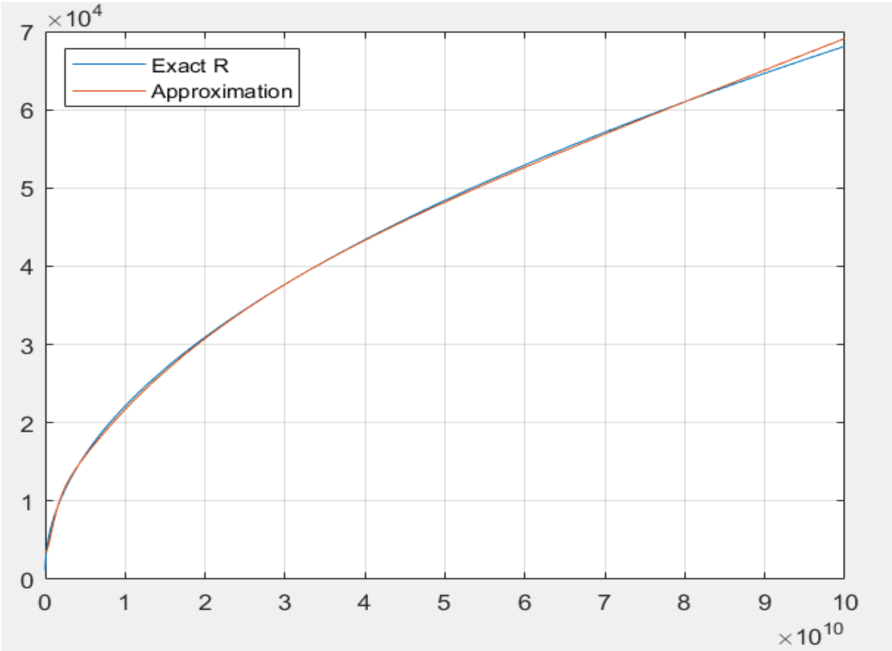

Figure 4.14: Shows the approximated rational function compared to the exact R(w).

2. **Direct Laplace-to-Time Domain Conversion:** The frequency-dependent resistance as in equation (102a) was inverse-transformed using equation(102b). This yielded equation(102c), but implementation in FDTD and RLC models failed to produce accurate results.

$$R(s) = R_o + R_s\sqrt{s}\,\sqrt{2} \qquad (102a)$$

$$\mathcal{L}^{-1}\{\sqrt{s}\} \;=\; -\frac{1}{2\sqrt{\pi}\ \ t^{3/2}}, where\ \mathcal{L}\{t^{-3/2}\} = -2\sqrt{\pi s} \qquad (102b)$$

$$R(t) = R_o + R_s - \frac{1}{2\sqrt{\pi}\ \ t^{3/2}}\sqrt{2} \qquad (102c)$$

These methodologies for integrating frequency-dependent resistance into time-domain models remain insufficient for THz applications. Further research is required to reconcile the frequency-dependent nature of R(w) with the time-invariant assumptions of RLC and FDTD. Promisingly, the Asymptotic Waveform Evaluation (AWE) method, discussed in the following section, demonstrates potential for synthesizing TL models that inherently account for R(w), offering a pathway to bridge this gap.

4.5 AWE testing

4.5.1 Impulse response:

Consider the following example:

$$A = \begin{bmatrix} -2 & 1 & 0 & 0 \\ 1 & -2 & 1 & 0 \\ 0 & 1 & -2 & 1 \\ 0 & 0 & 1 & -1 \end{bmatrix}, B = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \text{ } and \text{ } C = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{103}$$

One can find the impulse response by calculating moments, poles and residues using equations (51), (55) and (58) respectively,

$$m_0 = 1\,, m_1 = -4, m_2 = 30, m_3 = \ -246, m_4 = 2037, m_5 = -16886, m_6 = 140000$$

$$p_1 = -3.53, p_2 = -2.35, p_3 = -1, p_4 = -0.12$$

$$k_1 = 0.18, k_2 = 0.43, k_3 = 0.33, k_4 = 0.05$$

Then the impulse response is:

$$H(t) = \ 0.18e^{-3.53t} \ + \ 0.43e^{-2.35t} + 0.33e^{-t} + \ 0.05e^{-0.12t} \tag{104}$$

4.5.2 A theoretical method for validating AWE [12].

The impulse response of a state space model is given by:

$$y(t) = Ce^{at}B \tag{105a}$$

$$where, \quad e^{at} = Me^{\Omega t}M^{-1} \tag{105b}$$

$$and\ \Omega = M^{-1}AM = \begin{bmatrix} \lambda_1 & 0 & 0 \\ \dots & \lambda_2 & 0 \\ 0 & \dots. & \lambda_n \end{bmatrix} \tag{105c}$$

Considering the same example with $t = 0: 2$ with time step of 0.001 the following is obtained:

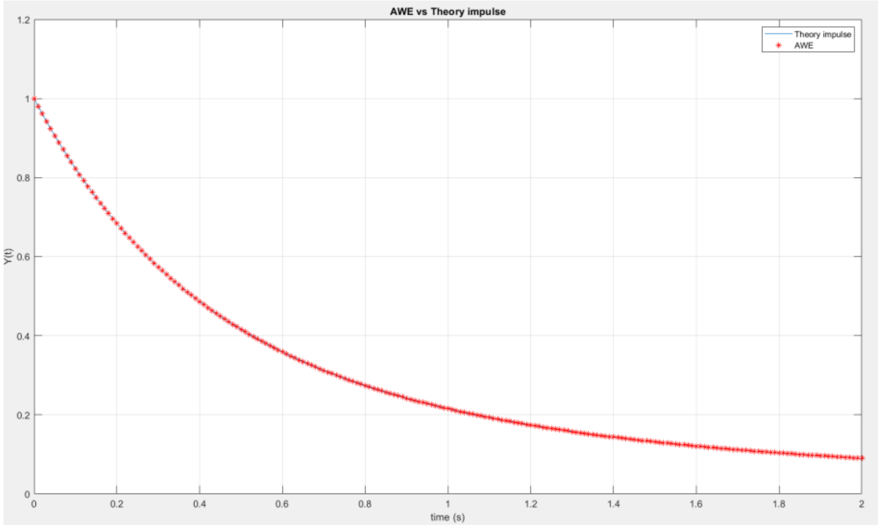

Figure 4.15: Comparison of AWE approximation and theoretical impulse response,

The Asymptotic Waveform Evaluation (AWE) method is tested using a system matrix approach, generating moments, poles, and residues to derive the impulse response function. The obtained RMSE value of $5.6523x10^{-8}$ indicates high accuracy, suggesting that AWE is a highly reliable method for transmission line analysis.

4.5.3-Unit step response.

Using the same example in (103),

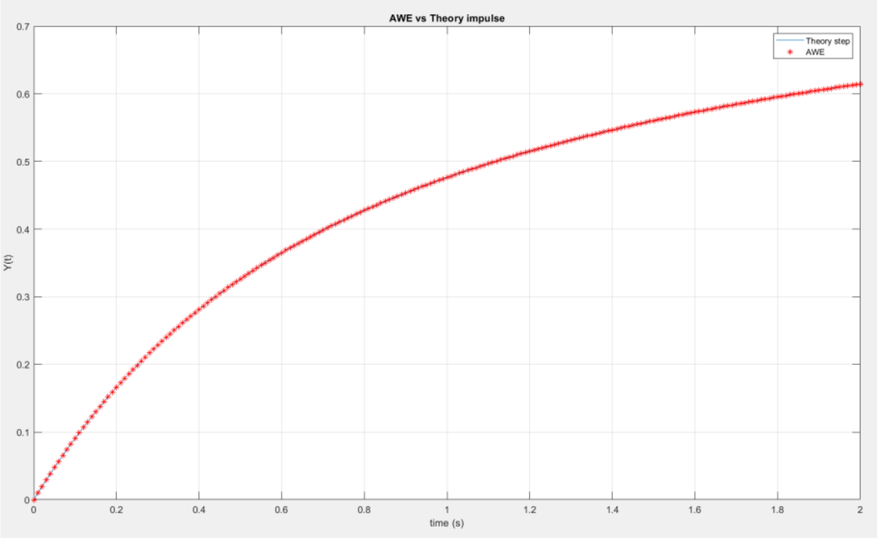

Figure 4.16: Comparison of AWE and the theoretical step response over time shows that AWE closely aligns with the theoretical model, highlighting its accuracy.

Similar to its impulse response, the unit step response of AWE also demonstrates exceptional accuracy, with an RMSE of $1.2973x10^{-8}$. The results highlight that reducing the time step further enhances AWE's accuracy, though at the cost of increased computational time.

4.6 Transmission Line Approximation Using Asymptotic Waveform Evaluation

This section implements the methodology described in Section 3.4 to construct an AWE model for transmission line analysis, utilizing Y-parameter measurements and step response for validation. The model is simulated using code 12 in the Appendix, configured with the following transmission line parameters:

$$C = 1x10^{-10}F, \qquad L = 250x10^{-9}H, \qquad R = 0.1\,Ohms, \qquad l = 400m,$$

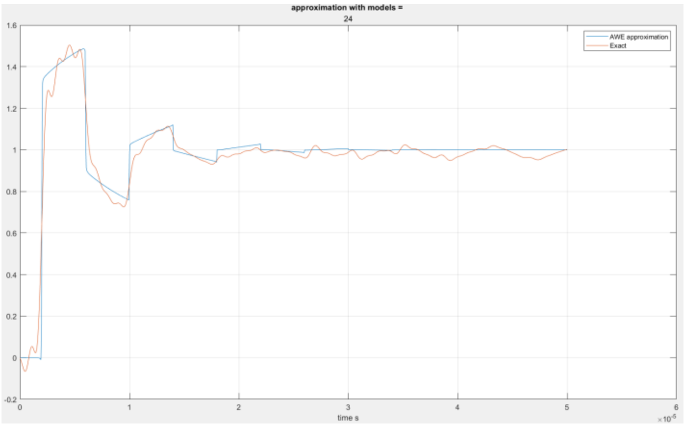

Figure 4.17: Shows the step response of the generated AWE model compared to the exact results.

Figure 4.13 illustrates the step response of the AWE model against exact solutions, demonstrating strong agreement with (RMSE) of 0.0869 (unit step) and 0.4269 (impulse response). However, during experimental validation, computational challenges arise when computing moments at terahertz (THz) frequencies due to numerical instability. To address this, the rank of the system matrix **A** is analysed. If the matrix rank is found to be lower than its dimension, adopting a first-order rational approximation—instead of the conventional second-order approach—effectively alleviates rounding errors and improves numerical accuracy. This adjustment ensures higher accuracy in modelling high-frequency behaviour at THz ranges, while maintaining computational reliability.

4.6.1 THz TL approximation that exhibit fractional-order frequency-dependent behaviour using AWE

Considering the parameters in (94) for THz TL, with the inclusion of R being frequency dependent and using code (13) in the appendix the following results were obtained. For the rational approximation, equation (79) is used for the resistance.

At THz frequencies, the selection of expansion points significantly influences the stability and accuracy of the AWE model. When expansion points are spread across a wide frequency range, the resulting poles often lack complex conjugate symmetry, which may compromise model fidelity. Empirical analysis revealed that expansion around zero frequency produces more stable and accurate approximations. In addition, experimental testing revealed that accurate modeling at these frequencies requires capturing key peaks in the frequency response, particularly those extending up to 7.5 THz. Including a greater number of these resonant features improves the approximation's ability to replicate the exact time-domain response, as illustrated in Figure (4.18).

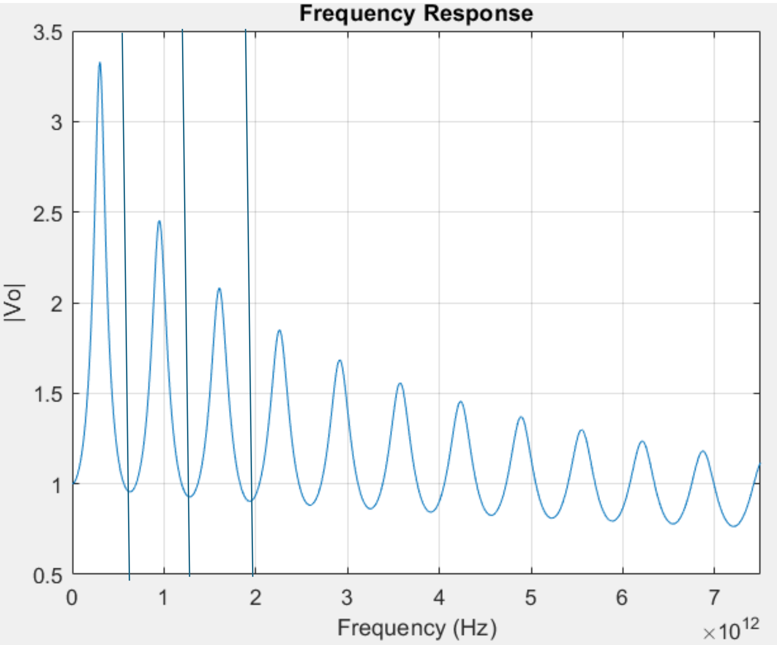

Figure 4.18: Shows the exact frequency response of TL with frequency dependent R up to 7.5 THz.

The approximation process aims to model these resonant peaks using the methodology presented in Section (3.4.1). Through extensive testing, it was determined that a model composed of 16 poles produced a high-fidelity step response, closely resembling the exact time-domain response, as shown in Figure (4.19). Through testing it's found that with 16 models promising accuracy is achieved, and the unit step response obtained as in Figure (4.19) compared to the exact solution.

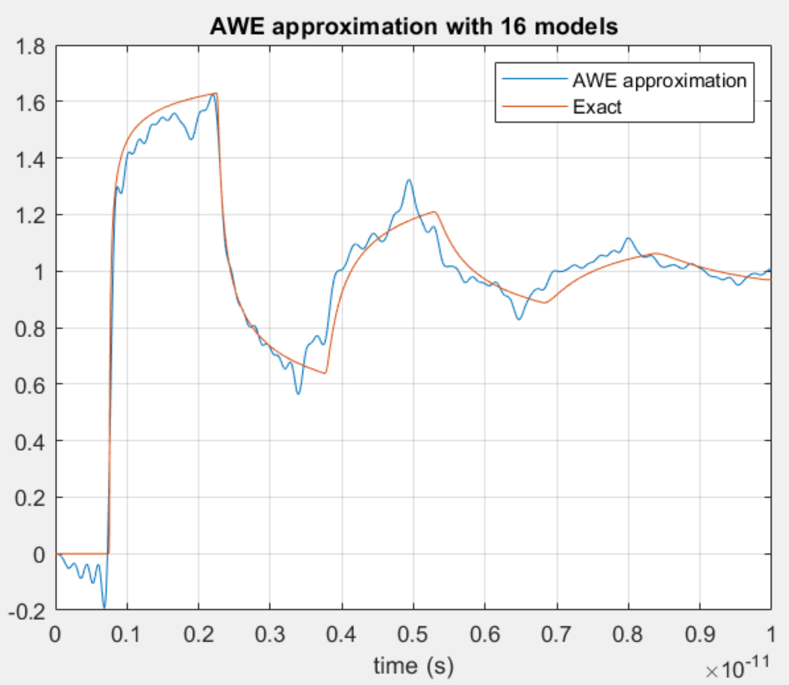

Figure 4.19: The unit step response of AWE approximation of the THz line compared to the exact model with R(w).

Figure 4.19 presents the unit step response obtained from the AWE model, demonstrating agreement with the exact solution. Furthermore, with 32 poles and corresponding residues, the root-mean-square error (RMSE) was calculated as 0.0630, reflecting high model accuracy. This performance surpasses conventional RLC and finite-difference time-domain (FDTD) models, which require discretization into more than 100 sections to achieve comparable accuracy, particularly when the frequency dependence of resistance is not considered, as shown in Table 4.

Unlike RLC and FDTD models, which are constrained by physical segmentation and time-stepping schemes, the AWE-based rational approximation offers a compact, parametric representation of the system (in the form of poles and residues). This form is readily adaptable to various inputs and different analysis scenarios, providing a versatile tool for high-frequency TL modelling.

4.6.2 AWE model response to Sinusoidal and Trapezoidal Inputs (with frequency-dependent resistance)

The unit step response derived through recursive convolution, as discussed in Section [3.3], is adapted to accommodate time-varying inputs such as sinusoidal and trapezoidal waveforms by modifying the recursive convolution approach. Two numerical strategies are implemented:

1. A straightforward approach is to assume that x(τ)≈x(t_{n-1}) is constant across [t_{n-1}, t_n] For sufficiently small Δt, this usually works well. Then, on each time step,

$$\int_{t_{n-1}}^{t_n} e^{p_i(\mathrm{t_n}-\tau)}x(\tau)\,\frac{\mathrm{d}\tau}{e^{p_i(\mathrm{t_n})}} \approx x(t_{n-1})e^{-p_i(\mathrm{t_n})}\int_{t_{n-1}}^{t_n} e^{p_i(\tau)}\,\mathrm{d}\tau = x(t_{n-1})\frac{1-e^{p_i\Delta t}}{-p_i} \qquad (106)$$

   Hence the update equation becomes

$$y_i(t_n) = e^{p_i\Delta t}y_i(t_{n-1}) + k_i\frac{1-e^{p_i\Delta t}}{-p_i}x(t_{n-1}) \qquad (107)$$

2. Using the same pole–residue recursive-convolution method but without approximating x(t) as a constant over each time step. The key is to compute the integral

$$\int_{t_{n-1}}^{t_n} e^{p(t_n-\tau)}\,\sin(\omega\tau)\,d\tau = e^{p_i\,t_n}\left[\frac{e^{-p_i\,\tau}\,(\,-p_i\,\sin(\omega\,\tau)\;-\;\omega\,\cos(\omega\,\tau)\,)}{(-p_i)^2\;+\;\omega^2}\right]_{\tau=t_{n-1}}^{\tau=t_n} \qquad (108)$$

   Hence the update equation becomes

$$y_i(t_n) = e^{p_i\Delta t}y_i(t_{n-1}) + k_i e^{p_i\,t_n}\left[\frac{e^{-p_i\,\tau}\,(\,-p_i\,\sin(\omega\,\tau)\;-\;\omega\,\cos(\omega\,\tau)\,)}{(-p_i)^2\;+\;\omega^2}\right]_{\tau=t_{n-1}}^{\tau=t_n} \qquad (109)$$

Code(14) and code(16) in the appendix apply the first approach and the second one respectively. The following results are obtained:

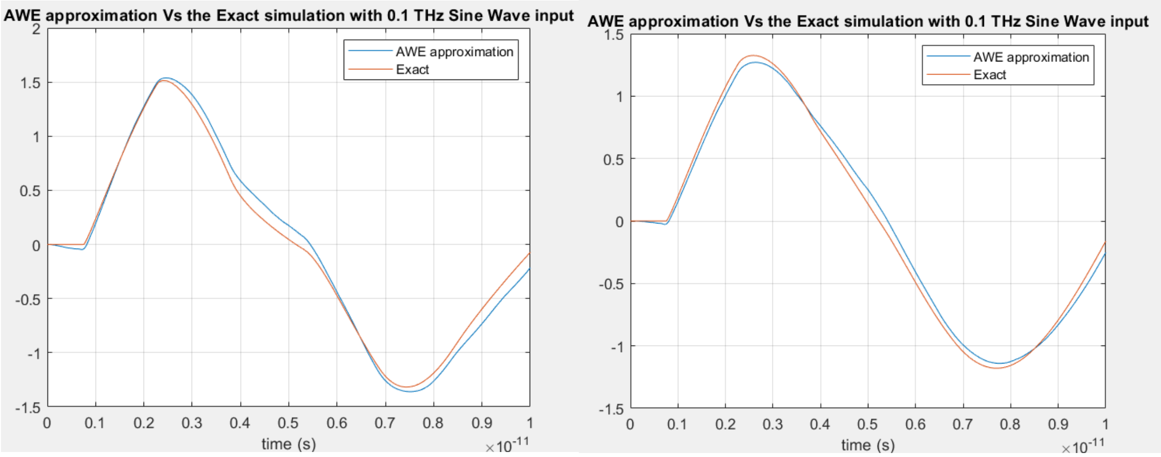

Figure 4.20: Shows the response of AWE model approximation of TL THz with Rs =0 and 10 compared to the exact model considering R(w).

Implementations of the second approach demonstrated RMSE values of 0.0881 (without the source resistance) and 0.0609 (when including Rs=10), Figure 4.20 contrasts the AWE model's response with Rs=0 and Rs=10 Ω against the exact solution.

To apply the recursive convolution approach to a trapezoidal pulse input x(τ), where x(τ) is piecewise-linear, it's assumed the time step Δt is sufficiently small to resolve the trapezoidal shape (i.e., multiple time points are used during the rise and fall phases). In this case, the simplest approach is to approximate x(τ) as either piecewise-constant or piecewise-linear over each small interval Δt. This can be achieved using numerical integration methods such as the Midpoint rule or Trapezoidal rule, as follows:

$$\text{avgInput} \approx \frac{x(t_{n-1}) + x(t_n)}{2}, \tag{110}$$

$$\int_{t_{n-1}}^{t_n} e^{p_i\,(\mathrm{t_n}-\tau)}\; x(\tau)\,\mathrm{d}\tau \approx [\text{avgInput}]\,\frac{1 - e^{\,p_i\,\Delta t}}{-p_i} \tag{111}$$

$$y_i(t_n) = e^{p_i \Delta t} y_i(t_{n-1}) + k_i(\text{avgInput})\,\frac{1 - e^{\,p_i\,\Delta t}}{-p_i} \tag{112}$$

Thus, by applying this method and implementing the associated algorithm (code (17) in the appendix), the following results are obtained:

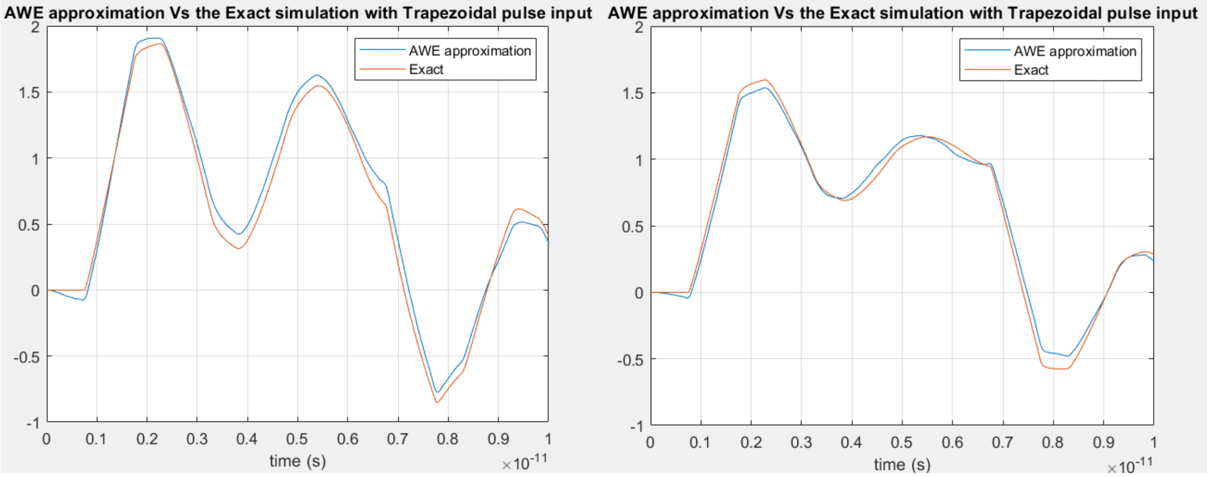

Figure 4.21: Comparison of the AWE approximation (blue) to an exact simulation (red) for a THz TL driven by a trapezoidal pulse.

The figure shows two time-domain responses of a TL excited by a trapezoidal pulse—one with Rs included and one without. In both cases, the AWE-based waveform (shown in blue) closely matches the exact solution (indicated by red wave). With Rs present, the root-mean-square error (RMSE) between AWE and the exact simulation is 0.0589, indicating tighter agreement. Without Rs, the RMSE increases slightly to 0.0922, reflecting more pronounced differences in amplitude or phase around the waveform’s transitions. Overall, these small RMSE values confirm that the AWE approximation, which uses pole–residue expansions to model system dynamics efficiently, provides an accurate yet computationally efficient method for transient response predictions.

4.6.3Challenges and Future Enhancements

1. **Pole Distribution**: Non-conjugate pole pairs observed at high-frequency expansion points may destabilize the model. Enforcing conjugate symmetry (e.g., replacing +a+jb with −a+jb) could improve accuracy.
2. **Curve-Fitting Constraints**: Ensuring exact DC response matching (via constrained curve-fitting) guarantees correct steady-state step response behaviour.
3. **Numerical Safeguards**: Rank-deficient matrices in rational approximation may necessitate first-order reductions, though this is less critical at THz frequencies due to sparse expansion points.

4.7 Rational approximation using curve fitting algorithm to obtain time domain TL THz model

Accurate time-domain modelling of transmission lines (TLs) requires rational approximation of their frequency-dependent parameters as discussed in the previous section. The Vector Fitting (VF) algorithm [18] and its Modified Vector Fitting (MVF) variant [19] provide a robust method for this purpose. This section details the numerical implementation of these algorithms for TL applications.

4.7.1 Mathematical Formulation

The frequency response of a TL parameter (e.g., propagation function h(s)) is approximated as a sum of partial fractions:

$$\mathrm{h}(\mathrm{s}) \approx \sum_{\mathrm{n}=1}^{\mathrm{N}} \frac{c_n}{s - a_n} + \mathrm{d} + \mathrm{s\,h}, \tag{113}$$

where $\{a_n\}$are poles, $\{c_n\}$ are residues, and d, h are real constants. The VF algorithm iteratively relocates initial poles $\{a'_n\}$ to optimal positions $\{a_n\}$by solving a sequence of linear least-squares (LS) problems. In addition, one can reformulate the problem by introducing an auxiliary function σ(s) such that:

$$\sigma(s)\,H(s) = \sum_{n=1}^{N} \frac{b_n}{s - p_n} + E(s), \tag{114}$$

where $\{p_n\}$ are some fixed "starting poles" (chosen by the user), $\{b_n\}$ are unknown coefficients, and E(s) includes unknown constant and linear terms. The function σ(s) is likewise represented in a partial-fraction-like form with the same fixed $\{p_n\}$. One then enforces the identity:

$$\mathrm{\sigma(s)\,H(s)} - \sum_{\mathrm{n=1}}^{\mathrm{N}} \frac{b_n}{s - p_n} = 0, \tag{115}$$

over many sample frequencies. Crucially, $\{p_n\}$ are held fixed in that step, making the problem linear (a simple overdetermined system in the unknown $\{b_n\}$ and in the parameters of E(s). After solving, one obtains a new function such that:

$$\sigma(s)\,H(s) = \frac{(\text{numerator})}{\prod_{n=1}^{N}(s - p_n)} \tag{116}$$

Then the zeros of σ(s) H(s) can be obtained, which become the updated pole set $\{a_n\}$. Additionally, re-inserting those new poles into the partial-fraction representation yields a refined fit for H(s). This two-step process is repeated in an iterative manner until convergence:

1. Solve a linear system for σ(s) H(s) with fixed $\{p_n\}$.
2. Update poles by computing the zeros of the fitted σ(s) H(s).

Once the final rational approximation is obtained with poles and residues, one can use the recursive method to obtain a time domain response.

4.7.2 Improvement to original vector fitting algorithm

The original Vector Fitting (VF) algorithm enforces a scaling function σ(s) to approach unity at high frequency, i.e. σ(s) → 1 as |s| → ∞. This constraint can be seen in the relation in equation (115) where the $\{p_n\}$ are fixed "starting" poles and the unknowns $\{b_n\}$ and E(s) are solved from measured or computed data. In practice, however, forcing σ(s)→1 can hamper the relocation of poles to the high-frequency region—particularly with noisy or insufficiently high-order data because it magnifies the least-squares penalty if σ(s) deviates strongly from unity. The Modified/Relaxed VF removes the strict σ(s)→1 requirement and replaces it with a milder summation constraint, such as:

$$\sum_{k} \mathrm{Re}\,\{\sigma(s_k)\} = \text{constant} \neq 0, \tag{117}$$

over the sampled frequencies $\{s_k\}$. This avoids trivial σ(s) = 0 solutions but still lets σ(s) vary more freely, improving pole relocation [19]. Numerical results confirm that this MVF converges faster, is far less sensitive to the initial pole placement, and remains effective even in the presence of noise, making it better suited for wideband or low-SNR rational approximations [19]. Moreover, MATLAB has this VF method implemented in a function called "rationalfit" over the sampled frequencies $\{s_k\}$ [20].

4.7.3 Development of a time-domain model from frequency-domain measurements using VF/MVF

The same THz transmission line parameters from Section 4.6 and code (18) provided in the appendix are used to derive the following responses.

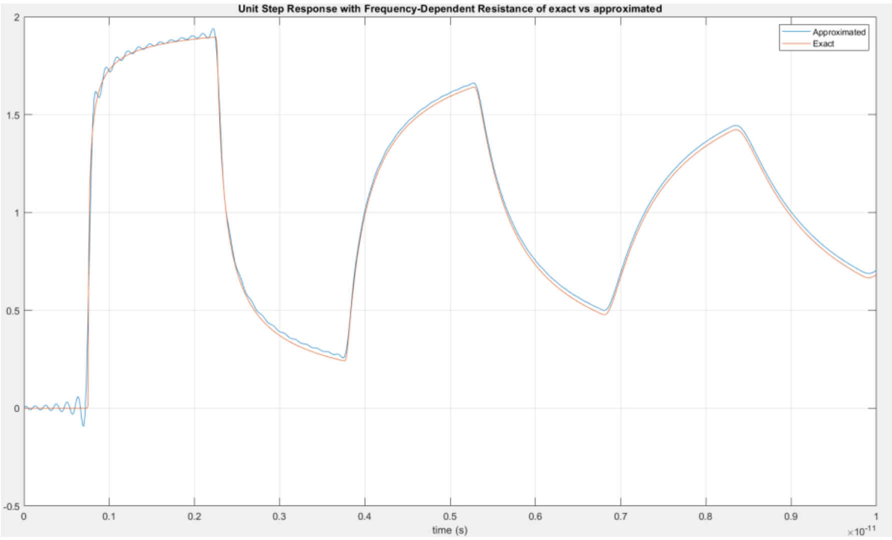

Figure 4.22: shows the unit step response with frequency-dependent R of the exact model compared to the approximation using VF/MVF With RMSE 0.0293 and 24 poles only

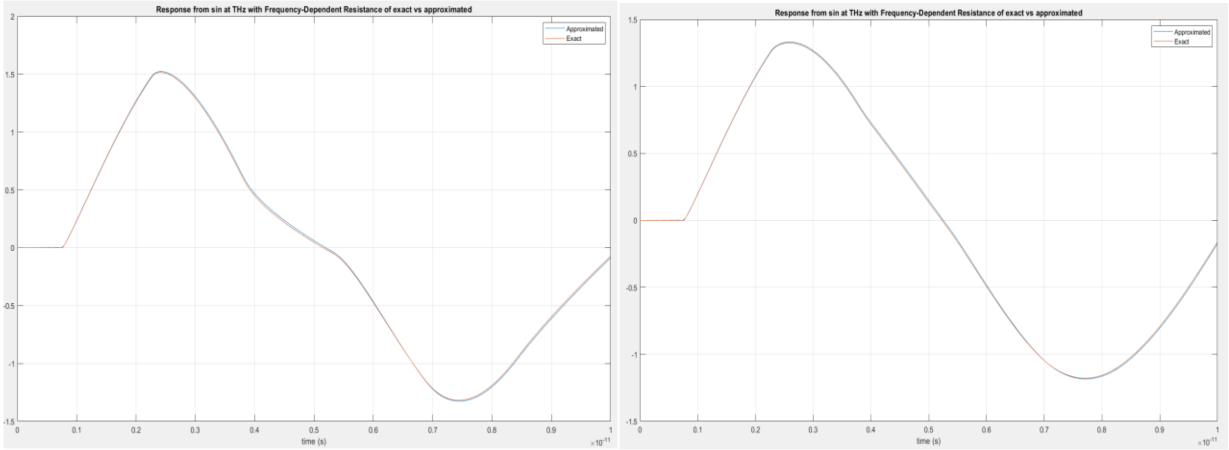

Figure 4.23: The results demonstrate the approximation accuracy of VF/MVF versus the exact model at 0.1 THz, with an RMSE of 0.0093 (including Rs) and 0.0119 (excluding Rs).

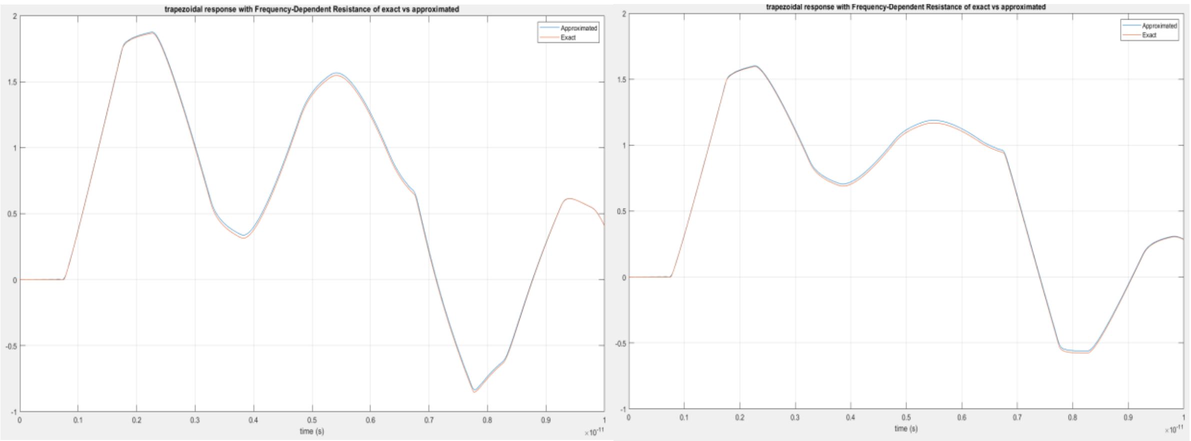

Figure 4.24: The VF/MVF approximation of the trapezoidal input response with frequency-dependent R shows an RMSE of 0.0130 (Rs = 10 included) and 0.0152 (Rs excluded) relative to the exact model.

The development of a time-domain model for the THz transmission line (TL) using the Vector Fitting (VF) and Modified Vector Fitting (MVF) algorithms demonstrates significant improvements in accuracy and computational efficiency. By approximating the frequency-dependent TL parameters as a sum of partial fractions, the VF/MVF approach iteratively relocates poles to optimal positions through linear least-squares optimization. This method achieved an RMSE as low as 0.0093 with only 24 poles, outperforming the AWE method from Section 4.6, which required 32 poles to attain an RMSE of 0.0630 for the unit step response.

Compared to the AWE-based curve fitting approach, the VF/MVF method offers superior accuracy and model compactness. For instance, while AWE achieved RMSE values of 0.0589–0.0922 for trapezoidal and sinusoidal inputs (depending on source resistance), the VF/MVF method consistently delivered lower errors (e.g., 0.0119–0.0293), even with fewer poles. This highlights the efficacy of VF/MVF's curve-fitting strategy over AWE's curve-fitting. Nevertheless, as discussed in Section 3.6, potential enhancements to AWE—such as enforcing conjugate pole symmetry and refining curve-fitting constraints—could significantly improve its ability to model THz transmission line behaviour, possibly bridging the performance gap with VF/MVF in future implementations.

4.8 Summery

This chapter evaluates the performance of FDTD, RLC ladder networks, and AWE/VF-MVF methods for transmission line analysis, revealing distinct trade-offs: FDTD improves accuracy with finer spatial discretization but incurs high computational costs (e.g., 0.5047 sec vs. RLC's 0.2088 sec at 400 sections), while RLC networks, leveraging adaptive ode23 solvers, achieve comparable accuracy (RMSE 3.5005 vs. FDTD's 3.2992) with superior efficiency. AWE excels in transient simulations, attaining ultra-low RMSE (down to $1.2973x10^{\{-8\}}$) for step/impulse responses but struggles with THz-frequency TL numerical instability, mitigated by rational approximations, whereas VF/MVF outperforms AWE in accuracy (RMSE 0.0093 vs. 0.0630) and model compactness (24 vs. 32 poles) through iterative pole relocation. Frequency-dependent resistance at THz challenges conventional methods due to factors such as skin effects, though AWE and VF-MVF partially address this via rational approximations. For trapezoidal/sinusoidal inputs, RLC and FDTD errors diminish with discretization at rising computational costs, while AWE/VF-MVF's recursive convolution methods achieve RMSE <0.0589, proving ideal for high-frequency, frequency-dependent systems. Overall, RLC suits rapid simulations, FDTD balances spatial resolution and speed, and AWE/VF-MVF dominate high-accuracy THz modeling, pending advancements in pole stability and dynamic parameter integration.

Chapter 5 – Ethics

Engineering ethics play a crucial role in the responsible development and implementation of technology. This project, which focuses on the simulation and exploration of terahertz (THz) transmission lines, adheres the ethical principles outlined in the IEEE Code of Ethics, the Engineers Ireland (IEI) Code of Ethics, and best practices in software usage, ensuring fairness, integrity, and professionalism throughout the research process.

6.1 IEEE Code of Ethics

The IEEE Code of Ethics [21] highlights the importance of integrity, safety, fairness, and professional responsibility in engineering practices. In alignment with these principles, the following ethical considerations were observed throughout this project:

- **Public Safety and Welfare:** The research was conducted with a focus on minimizing risks to public welfare, ensuring transparency in reporting findings, and considering the broader impact of THz technology.
- **Honest Representation of Results:** All data, simulations, and conclusions have been presented accurately, avoiding any misleading claims or misinterpretations.
- **Proper Credit and Acknowledgment:** Previous research, methodologies, and contributions from other professionals have been properly cited to maintain intellectual integrity.
- **Avoiding Conflicts of Interest:** The work was conducted objectively, with no external influence affecting the outcomes.

6.2 Engineers Ireland (IEI) Code of Ethics

As a professional engineering project, this work also follows the Engineers Ireland (IEI) Code of Ethics [22], which mandates professional responsibility and adherence to high ethical standards. The key IEI ethical principles applied in this project include:

- **Commitment to Public Safety and Environmental Protection:** The research was conducted responsibly, ensuring that findings contribute positively to engineering knowledge without causing harm.
- **Professional Competence and Integrity:** The project was executed within the researcher’s field of expertise, with clear communication of results and limitations.

- **Fairness, Courtesy, and Good Faith:** All collaborations and academic discussions were conducted respectfully and professionally, upholding ethical engineering standards.

6.3 Use of Licensed MATLAB and Software for Implementation and Testing.

In compliance with ethical engineering practices and software licensing agreements, this project exclusively used licensed versions of MATLAB and other required software for simulation, testing, and implementation [23]. This aligns with professional engineering standards, ensuring:

- **Legal and Ethical Software Usage:** Unauthorized or pirated software was strictly avoided, maintaining compliance with intellectual property laws.
- **Reliability and Accuracy of Results:** Using licensed software ensured that the simulations and calculations were performed using verified, up-to-date tools, minimizing errors and inaccuracies.
- **Multiple Testing for Accuracy:** The achieved results were tested multiple times using different parameter variations to ensure accuracy and reliability
- **Adherence to Industry Standards:** The project follows the best practices in software implementation to ensure high-quality, reproducible research outcomes.

By adhering to these ethical principles, this project upholds professional engineering standards, ensuring responsible and ethical research in THz transmission line modeling.

Chapter 6 - Conclusions and Further Research

This project evaluated the effectiveness of numerical methods for modelling terahertz (THz) transmission lines, focusing on balancing computational efficiency and accuracy. Our investigation focused on Finite-Difference Time-Domain (FDTD) methods, RLC ladder network simulations, and Asymptotic Waveform Evaluation (AWE) techniques. The results indicate that while FDTD offers excellent accuracy, its computational demands increase significantly with higher discretization levels requiring a very small-time step. In contrast, the RLC ladder network, particularly when leveraging adaptive solvers such as ode23, provides a favourable balance between computational efficiency and accuracy—especially when variable time steps are used.

The study further demonstrates that experimental measurements and testing showed the inclusion of frequency-dependent resistance, R(ω), is essential for accurately capturing transient responses at high frequencies (e.g., in the THz regime). However, the integration of R(ω) into time-domain models remains challenging, as conventional methods such as FDTD and RLC are inherently designed for time-invariant parameters. Addressing this limitation requires improved numerical techniques that seamlessly incorporate frequency-dependent losses while maintaining computational efficiency.

Several areas have emerged as promising avenues for future research. One important direction is complex Frequency hopping (CFH) for AWE, where optimizing expansion point selection in AWE using CFH could enhance stability and accuracy in high-frequency TL modelling. This approach may mitigate issues such as non-conjugate pole distributions and improve convergence when capturing resonant peaks up to THz ranges. Furthermore, Further development of AWE to enforce conjugate pole symmetry, refine curve-fitting constraints, and address numerical instabilities during moment calculations would strengthen its reliability for ultra-wideband simulations.

Another area of interest is the integration of frequency-dependent resistance R(ω) into RLC and FDTD models remains a crucial challenge. Future work should focus on incorporating advanced models for R(ω) into these simulations, enabling more accurate representations of high-speed circuit behaviours in THz applications.[redacted]These advancements will enable more efficient and precise modelling of next-generation high-speed circuits, addressing the growing demands of terahertz-frequency applications.

References

1. T. Kürner, "Turning THz Communications into Reality: Status on Technology, Standardization and Regulation," *IRMMW-THz 2018 - 43rd International Conference on Infrared, Millimeter, and Terahertz Waves,* Nagoya, Japan, 2018, pp. 1-2, doi: 10.1109/IRMMW-THz.2018.8510317.
2. X. Mou, Y. Chen, C. Ma, Y. Che, and J. He, “A numerical method to simulate THz-wave generation and detection of field-effect transistors,” in *Proc. IEEE,* 2008, p. 1, doi: 10.1109/IRMMW-THz.2008.2186.
3. D. M. Pozar, *Microwave Engineering*, 4th ed. Hoboken, NJ, USA: Wiley, 2012, pp. 48–50.
4. C. R. Paul, "Incorporation of terminal constraints in the FDTD analysis of transmission lines," in *IEEE Transactions on Electromagnetic Compatibility*, vol. 36, no. 2, pp. 85-91, May 1994, doi: 10.1109/15.293284.
5. Y. Shang, H. Yu and W. Fei, "Design and Analysis of CMOS-Based Terahertz Integrated Circuits by Causal Fractional-Order RLGC Transmission Line Model," in *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 3, no. 3, p. 355, Sept. 2013, doi: 10.1109/JETCAS.2013.2268948.
6. E. Gad, Y. Tao and M. Nakhla, "Fast and Stable Circuit Simulation via Interpolation-Supported Numerical Inversion of the Laplace Transform," in *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 12, no. 1, pp. 121-130, Jan. 2022, doi: 10.1109/TCPMT.2021.3122840.
7. Bandali, B., Gad, E., & Nakhla, M. (2022). Fast and Stable Transient Simulation of Nonlinear Circuits Using the Numerical Inversion of the Laplace Transform. IEEE Transactions on Components, Packaging, and Manufacturing Technology (2011), 12(7), 1171–1185. https://doi.org/10.1109/TCPMT.2022.3182136.
8. L. Brancik, "Matlab based time-domain simulation of multiconductor transmission line systems," *The IEEE Region 8 EUROCON 2003. Computer as a Tool.,* Ljubljana, Slovenia, 2003, pp. 464-468 vol.1, doi: 10.1109/EURCON.2003.1248066.
9. L. Brančík, "Numerical Inverse Laplace Transforms for Electrical Engineering Simulation," in MATLAB for Engineers – Applications in Control, Electrical Engineering,

IT and Robotics, K. Perutka, Ed. InTech, 2011, pp. 51-56. Available: https://doi.org/10.5772/2468.

10. W. T. Smith and S. K. Das, "Application of asymptotic waveform evaluation for EMC analysis of electrical interconnects," *Proceedings of International Symposium on Electromagnetic Compatibility*, Atlanta, GA, USA, 1995, pp. 429-434, doi: 10.1109/ISEMC.1995.523595.
11. M. Condon, *Lecturer Notes*, 2025.
12. M. Condon, *System Notes 2023*. Dublin, Ireland: Dublin City University, 2023, pp. 11–14.
13. W. T. Beyene and J. E. Schutt-Ainé, "Efficient transient simulation of high-speed interconnects characterized by sampled data," *IEEE Transactions on Components, Packaging, and Manufacturing Technology—Part B*, vol. 21, no. 1, pp. 105-114, Feb. 1998.
14. E. Chiprout and M. S. Nakhla, "Analysis of interconnect networks using complex frequency hopping (CFH)," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 14, no. 2, pp. 186-200, Feb. 1995.
15. E. Hairer, S. P. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations I: Nonstiff Problems*, 2nd ed. Berlin, Germany: Springer-Verlag, 1993, sec. II.4, pp. 164–168.
16. MathWorks, Inc., "ode23: Solve nonstiff differential equations—low order method," *MATLAB Documentation*, Mar. 28, 2025. [Online]. Available: https://www.mathworks.com/help/matlab/ref/ode23.html.
17. N. A. -Z. R-Smith, A. Kartci and L. Brančík, "Fractional-Order lossy transmission line with skin effect using NILT method," *2017 40th International Conference on Telecommunications and Signal Processing (TSP)*, Barcelona, Spain, 2017, pp. 730-734, doi: 10.1109/TSP.2017.8076084.
18. B. Gustavsen and A. Semlyen, "Rational approximation of frequency domain responses by vector fitting," in *IEEE Transactions on Power Delivery*, vol. 14, no. 3, pp. 1052-1061, July 1999, doi: 10.1109/61.772353.

19. B. Gustavsen, "Improving the pole relocating properties of vector fitting," in *IEEE Transactions on Power Delivery*, vol. 21, no. 3, pp. 1587-1592, July 2006, doi: 10.1109/TPWRD.2005.860281.

20. **MathWorks, Inc.** "rationalfit." *MATLAB Documentation*, 2025. [Online]. Available: https://uk.mathworks.com/help/rf/ref/rationalfit.html. [Accessed: 31-Mar-2025].

21. IEEE, "IEEE Code of Ethics," IEEE, Jun. 2020. Available: https://www.ieee.org/about/corporate/governance/p7-8.html.

22. Engineers Ireland, "Code of Ethics," Engineers Ireland, Feb. 2023. Available: https://www.engineersireland.ie.

23. MathWorks, “Policies and Statements.” Available: https://www.mathworks.com/company/aboutus/policies_statements.html. Accessed: Mar. 10, 2025.

Appendix

Code 1 (FDTD)

```
clear
clc
% a lossless , two-conductor line with Vs(t)=30,
% Rs = 0 R, VL(t) = 0, and RL = 100 R. The line is of
% length L = 400 m and has TJ = 2 x 10' m/s and ZC = 50 R
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
L_total = 400;  % Total length of the line (m)
Zc = 50;  % Characteristic impedance (Ohms)
v = 2e8;  % Speed of propagation (m/s)
Rs = 0;  % Source resistance (Ohms)
RL = 100;  % Load resistance (Ohms)
Vs = 30;%  % Input voltage (V)
% Compute inductance and capacitance
C = 1 / (v * Zc);
L = Zc/v;
NDZ = 20;  % Number of spatial steps
dz = L_total / NDZ;  % Spatial step delta z
dt = 1e-11;  % Time step delta t
t_max = 20e-6;  % Maximum simulation time (20 as in the paper)
t_steps = round(t_max / dt);  % Number of time steps
% allocate voltage and current arrays
V = zeros(NDZ+1, t_steps);
V(1,:)=Vs*ones(1,t_steps);
I = zeros(NDZ, t_steps);
% FDTD Loop for Time Stepping
for n = 1:t_steps-1
    V(1,n+1) = V(1,n);
    for k = 1:NDZ
    if k>1
        V(k,n+1) = V(k,n) + dt/(dz *C)* (I(k-1,n) - I(k,n));  % Update voltage
        I(k-1,n+1) = I(k-1,n)-(dt/(L*dz))*(V(k,n+1)-V(k-1,n+1));% Update current
    end
    end
    V(NDZ,n+1) =V(NDZ,n)+dt*(I(NDZ-1,n)/(C*dz)-V(NDZ,n)/(RL*C*dz));
end

% Plot the results for the voltage at the load
figure(1)
plot((0:t_steps-1)*dt/1e-6, V(NDZ,:));
xlabel('Time (\mus)');
ylabel('V Load (Volts)');
title('FDTD Simulation of Transmission Line');
grid on;
```

Code 2 (RLC)

```
clear
clc
len=400;
N = 20; % Number of sections in the transmission line
dz=len/N;
L = 2.5e-7*dz;   % Inductance
C = 1e-10*dz;    % Capacitance
R = 0;       % Resistance per section
```

```
Rs = 0;       % Source resistance
Rl = 100;     % Load resistance
Vs  = 30;      % Source voltage (could be a function of time)
y0 = zeros(2 * N, 1);
tspan = [0 20e-6];
% Solve using ode23 which shows better result in TL than ode45
[t, y] = ode23(@(t, y) fline(t, y, N, L, C, R, Rs, Rl, Vs), tspan, y0);
% Plot voltage at the end of the transmission line (VN)
figure(1);
plot(t, y(:,N*2));  % Plot voltage at the load (VN)
xlabel('Time (us)');
ylabel('Voltage at Load (V)');
title('RLC Voltage at Load over Time');
grid on
```

Code 3 (fline function)

```
function df = fline(t, y,N,L,C,R,Rs,Rl,Vs)
    % the use of t involves input dependent on time (i.e, sin(t))
    df = zeros(2 * N, 1);
    % Currents and voltages
    In = y(1:2:2*N);  % Currents
    Vn = y(2:2:2*N);  % Voltages
    % Boundary conditions at the source end (n=1)
    df(1) = (-1 / L) * Vn(1) - (Rs + R) / L * In(1) + (1 / L) * Vs;  % dI1/dt
    df(2) = (1 / C) * In(1) - (1 / C) * In(2);  % dV1/dt
    % Interior sections (n=2 to N-1)
    for n = 1:N-1
        df(2*n + 1) = (-1 / L) * Vn(n+1) - R / L * In(n)+(1 / L) * Vn(n);  % dIn/dt
        df(2*n) = (1 / C) * In(n) - (1 / C) * In(n+1);  % dVn/dt
    end
    df(2*N-1) = (-1 / L) * Vn(N) - R / L * In(N) + (1 / L) * Vn(N-1);
    df(2*N) = (1 / C) * In(N) - Vn(N) / (Rl * C);  % dVN/dt (voltage at the load)
end
```

Code 4 (NILTcv)

```
function [ft,t]=niltcv(F,tm,depict);
alfa=0; M=2506; P=3; Er=1e-10; % adjustable
N=2*M; qd=2*P+1; t=linspace(0,tm,M); NT=2*tm*N/(N-2);
omega=2*pi/NT;
c=alfa+log(1+1/Er)/NT; s=c-i*omega*(0:N+qd-1);
Fs(:,:,1)=feval(F,s); Fs(:,:,2)=feval(F,conj(s)); lv=size(Fs,1);
ft(:,:,1)=fft(Fs(:,:,1),N,2); ft(:,:,2)=N*ifft(Fs(:,:,2),N,2);
ft=ft(:,1:M,:);
D=zeros(lv,qd,2); E=D; Q=Fs(:,N+2:N+qd,:)./Fs(:,N+1:N+qd-1,:);
D(:,1,:)=Fs(:,N+1,:); D(:,2,:)=-Q(:,1,:);
for r=2:2:qd-1
 w=qd-r;
 E(:,1:w,:)=Q(:,2:w+1,:)-Q(:,1:w,:)+E(:,2:w+1,:);
 D(:,r+1,:)=-E(:,1,:);
 if r>2
 Q(:,1:w-1,:)=Q(:,2:w,:).*E(:,2:w,:)./E(:,1:w-1,:);
 D(:,r,:)=-Q(:,1,:);
 end
end
A2=zeros(lv,M,2); B2=ones(lv,M,2); A1=repmat(D(:,1,:),[1,M,1]);
B1=B2; z1=repmat(exp(-i*omega*t),[lv,1]); z=cat(3,z1,conj(z1));
```

```
for n=2:qd
 Dn=repmat(D(:,n,:),[1,M,1]);
 A=A1+Dn.*z.*A2; B=B1+Dn.*z.*B2; A2=A1; B2=B1; A1=A; B1=B;
end
ft=ft+A./B; ft=sum(ft,3)-repmat(Fs(:,1,2),[1,M,1]);
ft=repmat(exp(c*t)/NT,[lv,1]).*ft; ft(:,1)=2*ft(:,1);
switch depict
 case 'p1', plott1(t,ft); case 'p2', plott2(t,ft);
 case 'p3', plott3(t,ft); otherwise display('Invalid Plot');
end
```

```
% --- Plotting functions called by 1D NILT, vector version ----
%----------- Multiple plotting into single figure -------------
function plott1(t,ft)
figure; plot(t,real(ft)); grid on;
%figure; plot(t,imag(ft)); grid on; % optional
%----------- Plotting into separate figures ----------------
function plott2(t,ft)
for k=1:size(ft,1)
 figure; plot(t,real(ft(k,:))); grid on;
 figure; plot(t,imag(ft(k,:))); grid on; % optional
end
% ------------------ Plotting into 3D graphs ------------------
function plott3(t,ft)
global x; % x must be global in F
 m=length(t); tgr=[1:m/64:m,m]; % 65 time points chosen
 figure; mesh(t(tgr),x,real(ft(:,tgr)));
 figure; mesh(t(tgr),x,imag(ft(:,tgr)));
```

Code 5 (Exact solution with NILTcv)

```
clear
clc
R = 0;        % Resistance per unit length (Ohms per meter)
L = 2.5e-7;     % Inductance per unit length (Henries per meter)
G = 0;        % Conductance per unit length (Siemens per meter)
C = 1e-10;    % Capacitance per unit length (Farads per meter)
l = 400;    % Length of the transmission line (meters)
vs = 30;
t= 0:1e-11:20e-6;
h = t(2) - t(1);
% Calculate propagation constant (gamma) in the s-domain
z = @(s)(R+s.*L);
y = @(s)(G + s .* C);
gamma = @(s)sqrt(z(s) .* y(s));
% Calculate characteristic impedance (Z0) in the s-domain
Z0 = @(s) sqrt(z(s) ./ y(s));
% Calculate series impedance (Z_series) in the s-domain
Z_series =@(s) Z0(s) .* sinh(gamma(s) .* l);
Y_parallel =@(s) (1 ./ Z0(s)) .* tanh((gamma(s) .* l)./2); %y *tanh(gamma*l/2)
% Calculate parallel impedance (Z_parallel) in the s-domain
%Z_parallel =@(s) Z0(s) ./ tanh(gamma(s) .* l);
Z_parallel = @(s) 1./Y_parallel(s);
% Transfer function TF = Z_parallel / (Z_series + Z_parallel)
TF = @(s) Z_parallel(s) ./ (Z_series(s) + Z_parallel(s));
vo = @(s) TF(s) * vs./s;
```

```
%simplified
%vo = @(s) vs./(s.*cosh(l.*(G + C.*s).^(1/2).*(R + L.*s).^(1/2))); %
[y,t]=niltcv(vo,20e-6,'p1');
```

Code 6 (RLC to state space)

```
clear
clc
Rs =0;
Rdz = 5;
Ldz = 2;
Cdz =10;
N = 4;
numStates = 2 * N; % Each section has 2 states (current and voltage)
A = zeros(numStates, numStates);% Initialize A matrix
for i = 1:N
    if i == 1
        A(1, 1) = -(Rs + Rdz) / Ldz; % firs term (Rs + Rdz)
    else
        A(2*i-1, 2*i-1) = -Rdz / Ldz;
    end
    if i > 1
        A(2*i-1, 2*(i-1)) = 1 / Ldz;
        A(2*i-1, 2*i) = -1 / Ldz;
    end
    if i < N
        A(2*i-1, 2*i) = -1 / Ldz;
        A(2*i, 2*i-1) = 1 / Cdz;
        A(2*i, 2*i+1) = -1 / Cdz;
    else
        A(2*i, 2*i-1) = 1 / Cdz;
    end
end
% Initialize B matrix
B = zeros(numStates, 1);
B(1) = 1 / Ldz;
%C matrix
C = zeros(1, numStates);
C(end) = 1;
```

Code 7 (adapted FDTD for a lossy line)

```
function [y,t]=FDTD(R,L,C,length,Rs,t_max,N)
NDZ = N;  % Number of spatial steps
dz = length / NDZ;  % Spatial step delta z
dt = 1e-16;  % Time step delta t or Magic time step (dt = dz/v) for a lossles case
t_steps = round(t_max / dt);  % Number of time steps
% allocate voltage and current arrays
time = (0:t_steps-1)*dt;
V = zeros(NDZ+1, t_steps);
I = zeros(NDZ, t_steps);
f = 100e9;
% 1.Step input (1V source)
%Vs = 1 * ones(1, t_steps);
% 2. Sine wave (100 GHz)
Vs = sin(2*pi*f* time);
% 3. Trapezoidal pulse (custom function)
```

```
%for i=1:length(time)
    %Vs(i) = trapezoidalPulse(time(i));
%end
% FDTD Loop for Time Stepping
for n = 1:t_steps-1
     V(1, n+1) = (Rs*C/2*dz/dt+0.5)^-1*((Rs *C/2 *dz/dt-0.5)*V(1,n)-Rs*I(1,n)+0.5*(Vs(n+1)+Vs(n)));
    for k = 1:NDZ
    if k>1
        V(k,n+1) = V(k,n) + dt/(dz *C)* (I(k-1,n) - I(k,n));  % Update voltag
        I(k-1,n+1) = I(k-1,n)-(dt/(L*dz))*(V(k,n+1)-V(k-1,n+1))-(R*dt/L)*I(k-1,n);% Update current
    end
    end
    V(NDZ,n+1) =V(NDZ,n)+dt*(I(NDZ-1,n)/(C*dz));
end
y = V(NDZ,:);
t = time;
end
```

Code 8 (trapezoidal pulse function)

```
function v = trapezoidalPulse(t)
    % Define pulse parameters
    T_r = 1e-12;   % Rise time (1 ps)
    T_p = 5e-12;   % High-level duration (5 ps)
    T_total = 2*T_r + T_p;  % Total pulse duration (7 ps)

    % Piecewise definition of the pulse
    if t < 0
        v = 0;
    elseif t < T_r
        % Linear rise: from 0 to 1 V over T_r
        v = t / T_r;
    elseif t < T_r + T_p
        % Constant high level at 1 V
        v = 1;
    elseif t < T_total
        % Linear fall: from 1 V back to 0 over T_r
        v = (T_total - t) / T_r;
    else
        v = 0;
    end
end
```

Code 9 (AWE s=0)

```
function [h_impulse,h_s, y_step, t] = AWE(A,B,C,D,input,time)
    t = linspace(0,time,250);
    q = length(B);
    num_moments = 2 * q;
    moments = zeros(1, num_moments);
    [r,c]=size(C); % make sure C matrix in correct form
    if r~=1
        C= C';
    end
    for k = 1:num_moments
        moments(k) = (-1) * C * (A)^-(k) * B;
    end
    moments(1)=moments(1)+D;
    approx_order = length(B);
    % Construct the moment matrix
    moment_matrix = zeros(approx_order);
    Vector_c = -moments(approx_order+1:2*approx_order)';

    for i = 1:approx_order
        moment_matrix(i, :) = moments(i:i+approx_order-1);
    end

    % Solve for denominator coefficients
    b_matrix = moment_matrix^-1 * Vector_c;

    % Compute poles
    poles = roots([b_matrix', 1]);

    % Compute residues
    V = zeros(approx_order);
    for i = 1:approx_order
        for j = 1:approx_order
            V(i, j) = 1 / poles(j)^(i-1);
        end
    end

    A_diag = diag(1 ./ poles);
    r_moments = moments(1:approx_order);
    residues = -1 * (A_diag \ (V \ r_moments'));

    % Impulse response
    h_impulse = zeros(size(t));
    for i = 1:approx_order
        h_impulse = h_impulse + residues(i) * exp(poles(i) * t);
    end
    h_s =@(s) 0;
    for i = 1:length(poles)
        h_s =@(s) h_s(s) + residues(i)./(s-poles(i));
    end

    % Step response using recursive convolution
    y_step = zeros(size(t));
    y = zeros(length(poles), 1);

    for n = 2:length(t)
        dt = t(n) - t(n-1);
        exp_term = exp(poles * dt);
        for i = 1:length(poles)
            y(i) = residues(i) * (1 - exp_term(i))/(-poles(i)) * input + exp_term(i) * y(i);
        end
        y_step(n) = sum(y);
    end
end
```

Code 10 (generate (rational approximation) using frequency measurements)

```
function [H_impulse,num,deno]=generate_yp2(realV,imagV,wo)
Yr = realV;  % Real part of Y11
Yi = imagV; % Imaginary part of Y11
w = wo;
A = [];
C = [];
% Loop through each frequency point to construct A and C
A = [];
```

```
C = [];
% Loop through each frequency point to construct A and C
for k = 1:length(w)
    wk = w(k);
    Yr_k = Yr(k);
    Yi_k = Yi(k);
    % Construct rows for A and C
    A_row1 = [-1, Yr_k, 0, -wk*Yi_k]; % Real part
    A_row2 = [0, Yi_k, -wk, wk*Yr_k]; % Imaginary part

    % Append to A
    A = [A; A_row1; A_row2];
    % C
    C_row1 = wk^2 * Yr_k; % Real part
    C_row2 = wk^2 * Yi_k; % Imaginary part
    % Append to C
    C = [C; C_row1; C_row2];
end

% Solve for B = [a0; b0; a1; b1]
B = A \ C;
% get cof
a0 = B(1);
b0 = B(2);
a1 = B(3);
b1 = B(4);
num = [a1,a0];
deno = [1,b1,b0];
% generated H
H_impulse =@(s) (a1*s+a0)./(s.^2+b1*s+b0);
end
```

Code 11 (Generate state space model)

```
function [A,B,C,D]= create_state_space(nem,deno)
N = nem;
D = deno;
% Perform polynomial division to make it strictly proper
[Q, R] = deconv(N, D);
% check leading coefficient (assumed to be 1)
if D(1) ~= 1
    D=D/D(1);
    N = N/D(1);
    [Q, R] = deconv(N, D);
end
% Extract coefficients for state-space representation
g = D(2:end);  % Exclude leading coefficient ( g terms )
% f terms
if R(1) ==0
    f=R(2:end);
else
    f = R;
end
% Construct state-space matrices
n = length(g); % Order of system
A = [zeros(n-1,1), eye(n-1); -flip(g)];
B = [zeros(n-1,1); 1];
C = flip(f);
D = Q;
end
```

Code 12 (AWE TL model)

```
clear
clc
% generate 100 points.
f = 1e6;
f = linspace(0,f,100);
w = 2*pi*f;
s = i *w;
% exact solution and generating 100 points
vo =1./(cosh(400.*(0 + 1e-10.*s).^(1/2).*(0.1 + 2.5e-7.*s).^(1/2)));
% to be used in NILT
v =@(s)1./(s.*cosh(400.*(0 + 1e-10.*s).^(1/2).*(0.1 + 2.5e-7.*s).^(1/2)));
first_idx = 1:6;
time = 50e-6;
models =23;
[~,num,deno] = generate_yp2(real(vo(first_idx)),imag(vo(first_idx)),w(first_idx));
R0 = num(end)/deno(end);
[A,B,C,D] = create_state_space(num,deno);
%HAWE is Hs= @(s) resdue/s-pole + ...;1 is the inout, 50e-6 is t for plot
[~,HAWEi, y0] = AWE2(A,B,C,D,w(1),1,time);
N =4; % number of points per section or model
range = 6; % starting point of the second mmodel
for i=1:models
    range = range(end):N + range(end); % raange of frequency and exact values
    H_diff = vo(range)-HAWEi(s(range));
    [~,numi,denoi] = generate_yp2(real(H_diff),imag(H_diff ),w(range));
    %numi=numi*R0;
    %numi(end)=numi(end)*R0;
    [A,B,C,D] = create_state_space(numi,denoi);
    R0 = numi(end)/denoi(end);
    [~,HAWEj, yi, ti] = AWE2(A,B,C,D,w(range(1)),1,time);
    HAWEi = @(s) HAWEi(s)+HAWEj(s);
    y0 = y0+yi;
end
[y1,t1]=niltcv(v,time,length(y0));
RMSE = sqrt(sum(abs(y0-y1).^2)/length(y1)); %unit step
RMSE2 = sqrt(sum(abs(HAWEi(s)-vo).^2)/length(vo));% impulse
figure
plot(t1,y1,ti,y0)
%plot(f,abs(vo),f,abs(HAWEi(s)))
grid on
xlabel('time s')
legend('AWE approximation', 'Exact');
title('approximation with models = ',num2str(models+1));
```

Code 13 (curve fitting approximation at THz using AWE)

```
clear
clc
% Generate 100 frequency points for analysis
Ro = 1200;          % Resistance per unit length (Ω/m)
L = 250e-9;        % Inductance per unit length (H/m)
C = 1e-10;         % Capacitance per unit length (F/m)
Rs = 10;           % Source resistance (Ω)
G = 0;             % Conductance (S/m) - set to zero in this case
l = 150e-6;        % Length of transmission line (m)
f = 750e10;        % Maximum frequency (7.5 THz) to capture many peaks
f = linspace(1,f,100); % Create 100 frequency points from 1Hz to 7.5THz
w = 2*pi*f;
```

```
s = 1i * w;
% Exact solution calculation for comparison
% Frequency-dependent resistance including skin effect
R = Ro + 0.06*(1+1i)*sqrt(w);
% Exact voltage transfer function in frequency domain
vo = sqrt(s*C.*(R+s*L))./(sqrt(s*C.*(R+s*L)).*cosh(l*sqrt(s*C.*(R+s*L))) +
Rs*s*C.*sinh(l*sqrt(s*C.*(R+s*L))));
% Function definitions for NILT
% Skin effect resistance as a function of complex frequency
R = @(s) Ro + 0.06*sqrt(s)*sqrt(2);
% Voltage transfer function definition for NILT
v = @(s) sqrt(s*C.*(R(s)+s*L))./(sqrt(s*C.*(R(s)+s*L)).*cosh(l*sqrt(s*C.*(R(s)+s*L))) +
Rs*s*C.*sinh(l*sqrt(s*C.*(R(s)+s*L))));
v = @(s) v(s)*1./s; % Scale for unit step response

% Initial model fitting using first 7 frequency points
first_idx = 1:7;
time = 10e-12; % Simulation time window (10ps)
[~,num,deno] = generate_yp2(real(vo(first_idx)),imag(vo(first_idx)),w(first_idx));
[A,B,C,D] = create_state_space(num,deno);
% Perform  (AWE) for initial approximation
[~,HAWEi, y0,t1] = AWE2(A,B,C,D,w(1),1,time);
% approximation setup
N = 5; % Number of points per section/model
range = 7; % Starting point for second model
models = 15; % Total number of models to create
% approximation loop
for i = 1:models
    % Define frequency range for current model
    range = range(end)-2:N + range(end);

    % Calculate difference between exact solution and current approximation
    H_diff = vo(range) - HAWEi(s(range));

    % Fit new model to the residual error
    [~,numi,denoi] = generate_yp2(real(H_diff),imag(H_diff),w(range));
    [A,B,C,D] = create_state_space(numi,denoi);
    [~,HAWEj, yi, ti] = AWE2(A,B,C,D,w(1),1,time);

    % Update overall approximation
    HAWEi = @(s) HAWEi(s) + HAWEj(s);
    y0 = y0 + yi;
end

% Calculate exact solution using NILT for comparison
[y1,t1] = niltcv(v,time,1000);

% Calculate error metrics
R1 = sqrt(sum(abs(y0-y1).^2)/length(y1)); % RMS error for step response
R2 = sqrt(sum(abs(HAWEi(s)-vo).^2)/length(vo)); % RMS error for frequency response

% Plot results
figure(1)
plot(t1,y0,ti,y1) % Plot step responses
grid on
xlabel('time (s)')
ylabel('Response')
legend('AWE approximation', 'Exact');
title(['approximation with models = ',num2str(models)]);

% Alternative frequency response plot (commented out)
% figure(2)
% plot(f,abs(HAWEi(s)),f,abs(vo)); % frequency response
% grid on
% xlabel('Frequency (Hz)')
% ylabel('Magnitude')
% legend('AWE approximation', 'Exact');
```

Code 14 (adapted recursive method for sine input)

```
function [ y_sin,t]=sine_response(poles,residues,t)

dt = t(2) - t(1);
omega = 2*pi*100e9;
% Poles and residues (assumed known)
% e.g. poles = [...];  residues = [...];
% (User must supply these from the system analysis.)

% Allocate output arrays
y_sin = zeros(size(t));
y_state = zeros(length(poles), 1);

% Recursive convolution
for n = 2:length(t)
    dt = t(n) - t(n-1);
    exp_term = exp(poles * dt);

    % Input at previous time point (zero-order hold)
    input_val = sin(omega * t(n-1));

    for i = 1:length(poles)
        % Update the i-th state
        y_state(i) = ...
            y_state(i) * exp_term(i) + ...
            residues(i) * (1 - exp_term(i)) / (-poles(i)) * input_val;
    end

    % Sum contribution of all poles
    y_sin(n) = sum(y_state);
end
end
```

Code 15 (sine integral for recursive method)

```
function val = conv_sine_term(p, omega, t1, t2)
    % Returns integral of e^{p*(t2 - tau)} sin(omega*tau) from tau=t1 to tau=t2
    % Evaluate indefinite integral at the two boundary points:
    a = -p;  % because inside the integral we have e^{-p tau}
    b = omega;

    % Helper inline
    F = @(tau) exp(a*tau).*( a*sin(b*tau) - b*cos(b*tau) )/(a^2 + b^2);

    val = exp(p*t2) * ( F(t2) - F(t1) );
end
```

Code 16 (adapted recursive method for sine input using the exact sine integral)

```
function [ y_sin,t]=sine_response2(poles,residues,t)
f     = 100e9;             % Frequency of sine wave (Hz)
omega = 2*pi*f;
dt    = t(2) - t(1);
numPoles = length(poles);
y_sin = zeros(size(t));              % final output
y_i   = zeros(length(poles), 1);     % internal states for each pole
% main loop
for n = 2:length(t)
    dt = t(n) - t(n-1);
    for i = 1:length(poles)
        p_i = poles(i);
        % update y_i(n) = e^(p_i dt)*y_i(n-1) + residue * integral(...)
        y_i(i) = exp(p_i*dt)*y_i(i) ...
                 + residues(i)*conv_sine_term(p_i, omega, t(n-1), t(n));
    end
    % sum of all pole contributions
    y_sin(n) = sum(y_i);
end
end
```

Code 17(adapted recursive method for pulse input using midpoint approximation)

```
function [ y_pulse,t]=pulse_response(poles,residues,t)
% Assume you have poles, residues, and time vector t as before
y_pulse = zeros(size(t));           % total output
y_i     = zeros(length(poles),1);   % states for each pole

for n = 2:length(t)
    dt = t(n) - t(n-1);
    % Evaluate the trapezoid input at t(n-1) and t(n)
    x_in_lo = trapezoidalPulse(t(n-1));
    x_in_hi = trapezoidalPulse(t(n));
    x_avg   = 0.5*(x_in_lo + x_in_hi);

    for i = 1:length(poles)
        p_i = poles(i);
        % "previous state" factor
        y_i(i) = exp(p_i*dt) * y_i(i);
        % add the integral piece ~ x_avg * [ (1 - e^{p_i dt})/(-p_i) ]
        if abs(p_i) > 1e-30
            y_i(i) = y_i(i) + residues(i) * x_avg * (1 - exp(p_i*dt))/(-p_i);
        else
            % If p_i ~ 0, handle that limit carefully (approx dt)
            y_i(i) = y_i(i) + residues(i) * x_avg * (dt);
        end
    end
    y_pulse(n) = sum(y_i);
end
```

Code 18 Using “rationalfit” MATLAB function based on VK/MVK for THz modelling.

```
clear
clc
Ro = 1200;          % Resistance per unit length (Ω/m)
L = 250e-9;        % Inductance per unit length (H/m)
C = 1e-10;         % Capacitance per unit length (F/m)
Rs = 10;
G = 0;
l = 150e-6;        % Length of the transmission line
f_max = 750e10;
f = linspace(1,f_max,100);
w = 2*pi*f;
s = 1i*w;
t_max = 10e-12;
t = linspace(0,t_max,1000);
% trapezoidal pulse
Tr = 1e-12;  % 1 ps rise/fall
Tp = 5e-12;  % 5 ps high
Amp = 1;     % 1 V amplitude
wo = 2*pi*100e9; %0.1 THz sine input
vs_sine = @(s) wo./(s.^2 + wo^2);% Laplace transform of sin(wt)
% Laplace transform of the trapezoid
vpulse = @(s) (Amp./(Tr*s.^2)).*(1 - exp(-Tr.*s))- (Amp./(Tr.*s.^2)).*(exp(-(Tr+Tp).*s) - exp(-
(2*Tr+Tp).*s));
R = Ro+0.06*(1+1i)*sqrt(w); % frequency depndent R
% obtain the exact frequency mesurements
vo =
sqrt(s*C.*(R+s*L))./(sqrt(s*C.*(R+s*L)).*cosh(l*sqrt(s*C.*(R+s*L)))+Rs*s*C.*sinh(l*sqrt(s*C.*(R+s*L)
)));
% for NILT
R =@(s) Ro+0.06*sqrt(s)*sqrt(2);
vos =@(s)
sqrt(s*C.*(R(s)+s*L))./(sqrt(s*C.*(R(s)+s*L)).*cosh(l*sqrt(s*C.*(R(s)+s*L)))+Rs*s*C.*sinh(l*sqrt(s*C
.*(R(s)+s*L))));
vo_step = @(s) vos(s)./s;% exact step response
vo_sine = @(s) vos(s).*vs_sine(s);
vo_pulse = @(s) vos(s).*vpulse(s);
H = rationalfit(f,vo);
poles = H.A';
residues = H.C';
[y_step,t]=step_response(poles,residues,t);
[y_sine,t]=sine_response2(poles,residues,t);
[y_pulse,t]=pulse_response(poles,residues,t);
[y_NILT,t]=niltcv(vo_pulse,t_max,1000);
error=RMSE(y_pulse,y_NILT); %RMSE error
plot(t,y_pulse,t,y_NILT)
grid on
xlabel('time (s)')
title("trapezoidal response with Frequency-Dependent Resistance of exact vs approximated")
legend("Approximated","Exact")
```