Methods for Acceleration, and Stability Assessment of Single-rate and Multi-rate Electromagnetic Transient Simulations

by

Ajinkya Sinkar

A Thesis submitted to the Faculty of Graduate Studies of
The University of Manitoba
in partial fulfillment of the requirements of the degree of

Doctor of Philosophy

Department of Electrical and Computer Engineering
University of Manitoba
Winnipeg, Manitoba, Canada

University of Manitoba

THE UNIVERSITY OF MANITOBA
FACULTY OF GRADUATE STUDIES
****
COPYRIGHT PERMISSION

"Methods for Acceleration, and Stability Assessment of Single-rate and Multi-rate Electromagnetic Transient Simulations"

By

AJINKYA SINKAR

A Thesis submitted to the Faculty of Graduate Studies of
The University of Manitoba
in partial fulfillment of the requirements of the degree of

Doctor of Philosophy

Department of Electrical and Computer Engineering
University of Manitoba
Winnipeg, Manitoba

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners. I understand that my thesis may be made available to the public electronically.

Acknowledgments

First and foremost, I want to thank my PhD supervisor, Dr. A. M. Gole, for always being available for discussions and providing his thoughtful comments from time to time on my research work. I would also like to thank my colleague, Dr. Huanfeng Zhao, who helped me immensely with ideas and was always available for having stimulating discussions at every juncture during the course of my doctoral studies.

I would like to express my sincerest appreciation to my external examiner, Dr. J. R. Marti from the University of British Columbia as well as my Ph.D. advisory committee members, Dr. Yi Zhang, and Dr. Shaahin Filizadeh, for spending their valuable time reviewing my research work and providing their thoughtful comments.

I am also grateful for the financial support that I received from NSERC, the University of Manitoba, and Manitoba Hydro International during the course of my doctoral studies.

A big thank you to Dr. Anil Kulkarni, my M.Tech supervisor at IIT Bombay, who recommended that I pursue my doctoral studies in Dr. Gole's research group at the University of Manitoba. Also thanks to Dr. Mukesh Das from Electranix Corp., who was always available for discussions (technical as well as non-technical).

A special thanks to Dr. Ian Jeffrey from the University of Manitoba; I thoroughly enjoyed and learnt a lot from his two courses, the first on Advanced Matrix Algorithms, and the second on Parallel Processing.

Very grateful to two of my best friends, Dr. Ganesh Gore, and Ms. Vaibhavi Joshi for their continuous moral support during the course of my doctoral studies. Also, my friends in the lab, John Liu, Yue Yi, and Chen Jiang were always there to make the learning experience much more enjoyable.

Last but certainly not least, I want to thank my parents, my brother Aditya, my sister-in-law Nanda and my fiancée, Pallavi for their unconditional support and love.

Abstract

Electromagnetic transient (EMT) simulations of large power networks can be resource-intensive and prohibitively slow. One of the ways of accelerating EMT simulations is by exploiting the parallel processing capabilities offered by modern multi-core and multi-processor computing systems. Often, multi-rate simulation is advisable where different parts of the network are solved with different time-steps, reserving the smaller time-steps for sub-networks with fast transients and larger ones for the slower sub-networks. This thesis aims to investigate methods for speeding up single-rate EMT simulations of large power networks as well as develop a rigorous analytical approach for assessing the stability of multi-rate EMT simulations. In this regard, the work reported in this thesis is divided into two parts:

In the first part, we develop novel methods for accelerating single-rate EMT simulations using parallel processing as well as evaluate and compare the computational efficiency of some of the existing methods proposed in the literature.

Firstly, an alternative method for formulating the equations of a network for EMT simulations is presented. It uses descriptor state-space equations (DSE) to represent the dynamical equations of a circuit. A procedure for interfacing a DSE-based formulation with a companion circuits-based EMT simulator is also developed. This procedure enables the interfacing of arbitrary power networks with any commercial CC-based EMT simulation package and can also be used to speed up the simulation using parallel processing.

Subsequently, sparse matrix-based parallelization methods are used for speeding up DSE-based EMT simulations. In particular, two of the commonly used methods in the literature are used and their computational performance is compared. The first method transforms a sparse matrix into Block Diagonal (BD) form and can be used when the simulated network contains distributed parameter transmission lines and/or cables. On the other hand, the second one transforms a sparse matrix into a Bordered Block Diagonal (BBD) form and is more general in its application.

A novel universally passive delay-based interface is developed after this. It uses existing inductors in the circuit to partition the network in EMT simulations. It allows for simulation

speed up when the solution of the partitioned network is computed on a parallel computing platform. It is shown that the proposed interface has superior performance compared to other existing inductor-based partitioning approaches, and is guaranteed to be passive thus benefiting the numerical stability of the simulation.

In the second part of the thesis, a novel approach for the stability assessment of multi-rate EMT simulations of linear time-invariant (LTI) circuits is developed. Firstly, it is demonstrated that multi-rate EMT simulations can produce unstable results for stable continuous-time LTI circuits even when the well-known A-stable trapezoidal integration method is used. Further, it is shown that such simulations always yield a periodically varying system in the discrete-time domain. By exploiting this property and applying the well-known technique of 'lifting', a sampled data time-invariant representation for the simulated discrete-time system is obtained which can then be used for assessing the numerical stability of the simulation using eigenvalue analysis. The proposed method is useful for assessing the stability of multi-rate EMT simulations even before running them.

Contents

List of Figures

List of Tables

List of Acronyms

ac	Alternating Current
API	Application Programming Interface
CC	Companion Circuits
CPU	Central Processing Unit
dc	Direct Current
DSE	Descriptor State-space Equations
EMT	Electromagnetic Transient
FACTS	Flexible ac Transmission Systems
FPGA	Field Programmable Gate Array
GPGPU	General Purpose Graphics Processing Unit
HVdc	High Voltage dc
LCC	Line Commutated Converter
LTI	Linear Time Invariant
MATE	Multi Area Thevenin Equivalent
MMC	Modular Multi-level Converter
MNA	Modified Nodal Analysis
SPMD	Single Program Multiple Data
TS	Transient Stability
UPNP	Universally Passive Network Partitioning
VSC	Voltage Source Converter

List of Symbols

s	complex variable of Laplace transform
j	complex number equal to $\sqrt{-1}$
ω	angular frequency (*rad/sec*)
$\Re\{\}$	real part of a complex number
$\mathbf{X}$	a matrix
$\underline{x}$	a column vector
$\underline{0}$	a zero vector
$\mathbf{Z}(s)$	an impedance matrix in s domain
$\mathbf{Y}(s)$	an admittance matrix in s domain
$z(s)$	a scalar impedance function in s domain
$y(s)$	a scalar admittance function in s domain

Chapter 1

Introduction

This chapter presents the background material along with the literature review. The objectives of the thesis are also presented followed by its outline.

1.1 Background

Simulations have long been used by power engineers for analyzing the transient behavior of power systems. In the early days, power system simulation studies were performed using analog simulators, or using Transient Network Analyzers (TNA) that consisted of scaled-down analog models for various elements in a system [4,5]. However, due to the advancements in digital computing technology as well as its flexibility and low maintenance requirement compared to analog components, digital simulation programs are almost exclusively used nowadays [6]. A digital simulation program solves the differential equations (representing a system's transient behavior) on a computer using a numerical integration algorithm with a suitable time-step.

Traditionally, Transient Stability (TS) simulation programs have been used for studying slow transient phenomena like rotor angle stability in ac power grids [7]. In contrast, Electromagnetic Transient (EMT) simulation programs have been mostly used for studying fast transient phenomena like lightning and switching overvoltages, ferroresonance etc. [8].

TS-type programs make modelling simplifications by representing the electrical network equations in phasor domain. This permits the use of larger time-steps (in the range of a few milliseconds) for the simulations, which reduces their computational burden and makes them suitable for simulating large power networks. On the other hand, EMT simulations use detailed models for various network elements for which relatively smaller time-steps (in the range of a few microseconds) are required for ensuring that the simulation results are accurate.

Nowadays with the proliferation of power electronic equipment such as high voltage dc (HVdc) transmission and flexible ac transmission systems (FACTS) in power systems, EMT simulation, because of its point-on-wave property [8], is becoming the preferred tool for analysing the transient behavior of power systems. However, since EMT simulations also have a high computational burden, simulating large systems tends to be resource intensive and slow.

1.2 Electromagnetic Transient (EMT) Simulations

In the range of circuit simulation algorithms, EMT simulations offer the greatest level of detail in modelling various network elements (such as inductors, capacitors, transmission lines etc.). Most EMT simulators available today use the Companion Circuits (CC) approach (proposed by Dr. H. W. Dommel in 1969 [9]) for formulating the equations that govern the behavior of a given power network.

In this approach, each branch inductor or capacitor is firstly transformed to their corresponding discrete time companion circuits using the trapezoidal integration method. As shown in Fig. 1.1, these companion circuits consist of a history current source, which is a function of the element's current and voltage in the previous time-step, in parallel with a conductance which depends on the time-step Δt used for simulation (the expressions for $I_{hL}(t-\Delta t)$ and $I_{hC}(t-\Delta t)$ seen in Fig. 1.1 can be found in [8]). After this, the difference equations of the circuit are formulated using Modified Nodal Analysis (MNA) [10] in the

form given in (1.1).

$$\mathbf{Y}\,\underline{V}(t) \quad = \underline{J}(t) \tag{1.1}$$

$$\Longrightarrow \underline{V}(t) = \mathbf{Y}^{-1}\underline{J}(t) \tag{1.2}$$

Here,

$\mathbf{Y}$: Augmented admittance matrix of the discretized circuit.

$\underline{V}(t)$: Vector containing node voltages, and voltage source branch currents at time t.

$\underline{J}(t)$: Vector containing history sources, current sources, and voltage sources at time t. Eq. (1.1) is then solved at every time-step to obtain the transient response of a network. Note that even though $\mathbf{Y}^{-1}$ is shown in (1.2) for solving (1.1), $\mathbf{Y}$ is seldom inverted in practice. Instead a factorization method such as LU decomposition, which is computationally more efficient than direct matrix inversion [11], is used for solving (1.1).

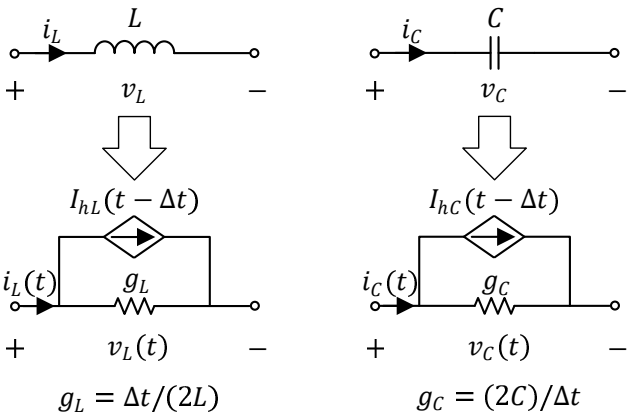


Figure 1.1: Companion Circuits for Inductor and Capacitor

The size of $\mathbf{Y}$ is directly proportional to the system's size. This means solving (1.1) can become computationally intensive as the system size grows. Moreover, when power electronic converters are present in the system, the power electronic switches (making up the converters) may change their conduction state very frequently. Additionally, each time a switch changes its conduction state (OFF to ON or ON to OFF), $\mathbf{Y}$ needs to be refactorized in order to solve (1.1). Also, smaller time-steps, in the range of a few nanoseconds to microseconds, are required for accurately simulating fast switching modern power electronic systems like

voltage source converters and modular multilevel converters [12].

All these reasons can make EMT simulations of large systems computationally intensive and sometimes prohibitively slow. Therefore, there is a need to develop methods that can speed up EMT simulations so that they can handle large systems efficiently.

1.3 Existing Methods for Speeding Up Transient Simulations of Power Systems

Many methods have been proposed in the literature for making large system simulations computationally efficient. One of the ways is to use a hybrid simulation approach which combines the benefits of two or more type of solvers [13]. The system to be simulated is divided into a smaller-sized *study zone* (where the disturbance occurs) and a larger *external system* (containing the rest of the system). The *study zone* is simulated using a detailed solver while the *external system* is simulated using a less detailed but computationally efficient solver. The rationale behind this approach is that usually in large systems, the area of interest around the point of disturbance is small as fast transients do not propagate over long distances [13].

An EMT-TS hybrid simulation approach was first proposed in [14]. Over the years, other hybrid simulation approaches having better accuracy performance have also been proposed which include the EMT-Dynamic Phasor (DP) approach [15] as well as EMT-DP-TS approach (where the DP simulation engine acts as a buffer layer between the more detailed EMT simulation and the less detailed TS simulation programs) [16]. However, few of the key challenges that exist for any hybrid simulation approach are:

1. Interfacing of different solvers with high accuracy and numerical stability.

2. Inevitable trade-offs while selecting the location of the boundary buses between the *study zone* and the *external systems*:

    - A boundary bus too close to the point of disturbance may give rise to inaccuracies

and numerical instability problems.

- A boundary bus too far away from the point of disturbance offsets the computational gains accrued by using a hybrid simulation approach as the portion of the network simulated using a detailed solver becomes larger

An alternative way of speeding up large system simulations is through the use of dynamic system equivalents [17]. The idea of dividing the system into a *study zone* and an *external system* is still used in this case. However, the *external system* is now modelled using a reduced order system equivalent. Various methods for deriving dynamic system equivalents that are accurate in the low frequency (i.e., a few Hz) and/or the high frequency (i.e., a few kHz) range are reviewed in [17].

Frequency Dependent Network Equivalents (FDNE) approximately model the measured (or the simulated) terminal admittance frequency response of an *external system* in the high frequency range by using either rational transfer functions [18] or lumped parameter models [19]. The computational efficiency of FDNEs for higher order systems can be further improved by using a Two Layer Network Equivalent (TLNE) [20,21]. In contrast, for deriving low frequency equivalents, aggregation methods based on coherency [22] or mode shapes from small-signal analysis [23] have been typically used. A wideband equivalent that can model the behaviour of an *external system* both in the low frequency and high frequency range has also been developed in [24]. It essentially uses a hybrid EMT-TS simulation approach with a multiport FDNE connected at the boundary buses. The TS simulation models the low frequency behavior while the multiport FDNE models the high frequency behavior of the *external system*.

The use of dynamic system equivalents does lead to significant computational gains (as is evident from the results reported in the current literature [25]). However, some of the major challenges with using these methods include:

1. The requirement to re-derive an equivalent from scratch if the *external system* undergoes any topological change close to the boundary bus [26].

2. The need for separate passivity enforcement strategies to ensure that the equivalent is passive in the frequency range of interest [27, 28].

3. The need for trade-offs between accuracy and passivity of a reduced order equivalent.

4. No access to internal variables of the *external system* (like node voltages, branch currents etc.) when it is replaced with a reduced order equivalent.

In recent years, considerable research effort has been directed towards speeding up EMT using parallel processing techniques. Earlier applications of parallel processing for accelerating EMT simulations relied on the use of expensive hypercube machines [29], and computer clusters [30]. However, with the development of multi-core CPUs (i.e., multiple processing units on a single integrated circuit) and other hardware-based accelerators (such as GPGPUs and FPGAs), the cost of enabling parallelism in general purpose applications is going down [31]. These modern high performance computing devices have also been used for speeding up EMT simulations as shown in [32–35].

The first step in the development of any parallel processing application is to partition the system into multiple subsystems [36]. In the next section, we will review the state-of-the-art methods used for partitioning an electrical network in EMT simulations so that parallel processing capabilities can be used for speeding up the simulation.

1.4 State-of-the-Art Partitioning Methods for Enabling Parallel EMT Simulations

Developing parallel EMT simulations firstly requires that the simulated system be divided into multiple subsystems. This is done so that each subsystem can then be solved in parallel in every time-step on a separate computing node. The solution of the entire system of course needs to be reconciled before proceeding to the next time-step. In this section, we review various methods that are currently used for partitioning a system for enabling parallel EMT simulations.

1.4.1 Network-level Partitioning Methods

Network-level partitioning methods decompose the original network into multiple sub-networks following which the equations of each sub-network are formulated separately. To partition a system at network-level, EMT program developers have relied on different types of interfaces and partitioning techniques. Various interfaces and partitioning techniques that have been traditionally used for parallel EMT simulations can be classified as: 1) delay-based partitioning, and 2) diakoptics-based partitioning.

Delay-based Partitioning

Delay-based partitioning relies on the finite time-delay introduced by a component for partitioning the system. This delay may be actual, such as the one arising from the finite transport delay across a transmission line, or it may be artificial, where signals in the present time-step are transferred to the remote system in the next time-step. When two or more subsystems in an EMT simulation are connected only using a delay-based interface as shown in Fig. 1.2, the augmented admittance matrix ($\mathbf{Y}$) becomes block-diagonal (also shown in Fig. 1.2). This means each block matrix $[Y_i]$ can be factorized separately and for every $i = 1, 2 \ldots n_{blocks}$, the $[Y_i]\,[V_i] = [J_i]$ can be solved independently within a time-step. Thus, the subsystems are completely decoupled for computations within a time-step, thereby enabling parallelism (the coupling between the various subsystems is through the right hand side vector $\underline{J}(t)$).

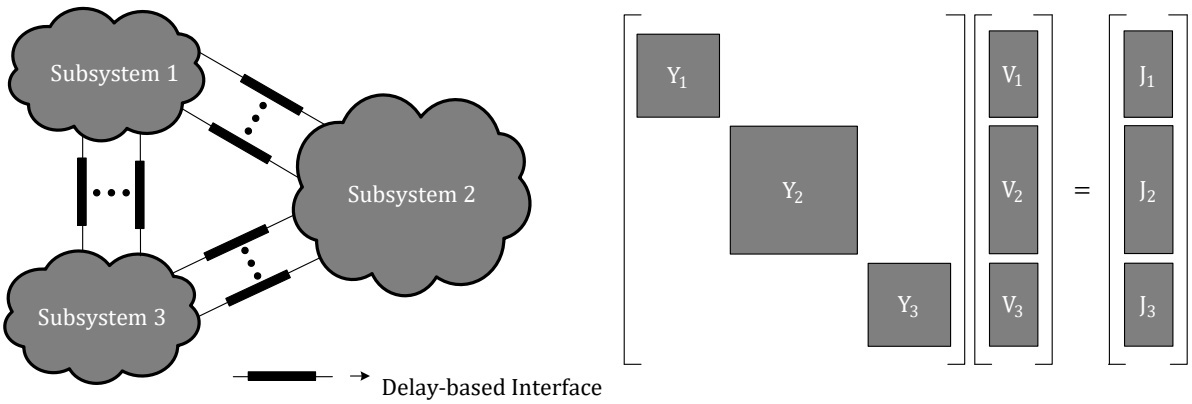


Figure 1.2: Block-diagonal Y matrix with delay-based interfaces

The well-known technique which uses distributed parameter transmission lines for partitioning comes under this category [29, 37]. If the travel time of the distributed parameter line is greater than one time-step, then the two sides are decoupled as information from one side would be available to the other side after the travel time has elapsed. There is no loss of accuracy in this case as the transport delay is real.

However, when no transmission line is available at the point of partitioning, previous researchers have proposed to replace inductors and capacitors at the point of partitioning with history sources [3, 38]. Representing an inductor or capacitor by a history source introduces a one time-step delay which can be used for partitioning. The rationale behind this is that inductor currents and capacitor voltages are continuous functions of time and do not change abruptly in a time-step. However, it can be shown that such methods may give rise to numerically unstable simulations in certain cases (this is demonstrated in Chapter 4).

An alternative approach has been to use a fictitious "stub-line" with a one time-step travel-time to connect systems together [39, 40]. When an inductor is present at the point of connection, its value can be incorporated into the Bergeron stub-line data via inclusion of additional (fictitious) shunt capacitance [39]. The stub-line interface is inherently passive and hence does not give rise to numerical instability in simulations. However, it exhibits sharp poorly damped resonance peaks in the high frequency range which can hamper its accuracy and can give rise to spurious high frequency oscillations in certain cases (as will be shown in Chapter 4).

Diakoptics-based Partitioning

In diakoptics-based partitioning, the system is firstly torn apart to get multiple subsystems [41]. Each subsystem is then represented using a multiport Thevenin/Norton equivalent after collapsing its internal nodes and it is connected to other subsystems through its boundary nodes. Subsequently, the solution for the entire system is obtained in every time-step using the following two-step procedure:

1. Compute the solution of the boundary system formed by the multiport Thevenin/Norton

equivalents,

2. Parallelly compute the solution of the equations that represent the internal details of each subsystem using the boundary node voltages/currents found in Step 1 as inputs for each subsystem.

A flowchart of the simulation when using this partitioning technique is shown in Fig. 1.3. Several diakoptics-based partitioning approaches for EMT simulations have been proposed in literature [30, 42, 43].

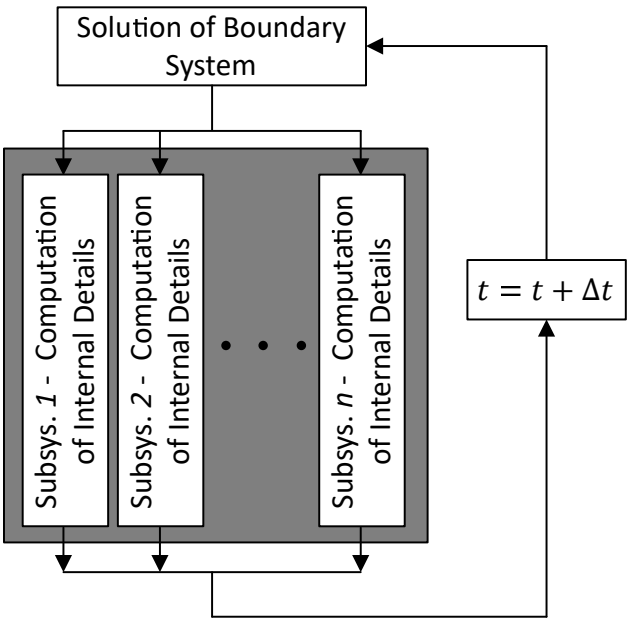


Figure 1.3: Flowchart of EMT simulation with diakoptics-based partitioning

The main difference between individual techniques is in the following:

1. The type of multiport equivalent that is used for each sub-system (Thevenin equivalent [30] or Norton equivalent [42]).

2. The approach used for formulating the equations which represent the internal details of each subsystem (companion circuits approach [30] or state-space approach [43]).

A diakoptics-based partitioning strategy is mathematically exact (i.e., gives the same accuracy as doing an unpartitioned solution) and facilitates partitioning at any arbitrary point

in the system (i.e., it does not rely on any delays introduced by a component in the system). However, since computing the boundary system's solution is an inherently sequential step, significant computational gains can only be accrued if the size of the boundary system is relatively small compared to the the size of each subsystem.

1.4.2 Matrix-level Partitioning Methods

For large power networks, $\mathbf{Y}$ in (1.1) is typically very sparse. However, the non-zero pattern of $\mathbf{Y}$ is not directly evident at the network-level. Due to this, it is challenging to ensure an equitable sharing of the computational load when using network-level partitioning methods. Alternatively, matrix-level partitioning methods use graph-based algorithms for decomposing the $\mathbf{Y}$ matrix such that parallel processing can be exploited for speeding up the solution of (1.1). In these methods, instead of partitioning the original circuit at the network-level, the equations of the entire circuit are firstly formulated in the form given in (1.1). Following this, an undirected graph corresponding to the non-zero pattern of $\mathbf{Y}$ is obtained. Finally, using different graph-partitioning algorithms [44, 45], $\mathbf{Y}$ is converted into forms which are amenable to parallel processing.

Several matrix-level partitioning methods have been proposed in the literature. In one of the methods, $\mathbf{Y}$ is transformed to a block diagonal form [46, 47] if the simulated network uses distributed parameter transmission lines and/or cable models. Parallelization of the solution with a block diagonalized $\mathbf{Y}$ can then be achieved in the same way as in the case of delay-based interfaces discussed in Section 1.4.1. An alternative to this is to transform $\mathbf{Y}$ to a bordered block diagonal form and then parallelize the simulation by solving a block lower triangular and a block upper triangular system as proposed in [34]. More recently, an alternative domain decomposition method (called the Link Domain Extraction Decomposition) has also been proposed in [48]. In this method, $\mathbf{Y}$ is expressed as a sum of a block diagonal matrix (BDM) and a linking-domain matrix (LDM). Following this, the Woodbury's formula [49] is used for computing the inverse of $\mathbf{Y}$ in order to solve (1.1) in parallel.

Although various matrix-level parallelization methods have been proposed in the liter-

ature for accelerating EMT simulations, a quantitative comparison of their computational performance has not been done.

1.5 Multi-rate EMT Simulations

Power systems are known to consist of "fast" and "slow" components that require widely different time-steps for accurately modelling their transient behavior in a simulation [50]. For example, for networks containing only traditional power system components (e.g., synchronous machines), experience shows that a time-step of about 25-50 μs is sufficient for obtaining accurate results; whereas for accurately simulating power electronic converters like voltage source converters, a much smaller time-step of 1-2 μs (25-50 times less than that for traditional power systems) is generally required. If a single time-step Δt is used for discretizing all the components, there is a potential for the simulation to become excessively slow as the Δt, selected based on the accuracy requirements of the fastest component in a system, is also used for solving relatively slower portions of the system.

One of the ways proposed in the literature for enhancing the computational efficiency (and hence the speed) of EMT simulations is to divide the network into multiple sub-networks and then use disticnt time-steps for simulating each sub-network. These are referred to as *multi-rate* EMT simulations. The time-step for simulating each sub-network is selected based on their individual time response characteristics [51], and the solution for the entire system is reconciled every time the simulation time-grid for the various subsystems coincides with each other. Multi-rate simulations are very useful in reducing the computational burden of the simulated networks in real-time simulations where the simulation time must be synchronized with the real-world clock [52].

One of the first approaches for multi-rate EMT simulations is presented in [50] which uses frequency-dependent transmission lines as the interface point for segmenting a system into fast and slow subsystems. A similar approach is used for multi-rate interfacing with a Bergeron transmission line model in the real-time simulator from RTDS Technologies Inc. [52]. Over the years, several other approaches have also been developed for carrying out

multi-rate EMT simulations e.g., [53,54]. In [2,55], an extension of the Multi Area Thevenin Equivalent (MATE) concept is proposed for carrying out multi-rate EMT simulations on the University of British Columbia's real-time simulator (called the Object Virtual Network Integrator [30]).

Although several publications have successfully demonstrated the computational efficiency enhancement accrued from using multi-rate interfacing techniques in EMT simulations, the numerical stability properties of various multi-rate EMT simulation algorithms have not been examined thoroughly.

1.6 Objectives of the Thesis

The main objectives of this thesis are to develop novel methods that help in partitioning the network in order to accelerate EMT simulations using parallel processing, and to analyze the stability and computational efficiency of the existing methods. To that end, the thesis aims to do the following:

1. Develop and investigate methods that enable network-level partitioning for accelerating EMT simulations.

   (a) Develop a diakoptics-based partitioning approach that uses Descriptor State-space Equations (DSE) for representing the internal details of each subsystem in partitioned EMT simulations.

   (b) Develop a novel delay-based interface that is highly accurate, does not compromise the numerical stability of the simulation, and helps in partitioning the system in EMT simulations using inductors that are already present in the network.

2. Develop an approach for assessing the numerical stability of multi-rate EMT simulations.

3. Compare the computational performance of existing matrix-level parallelization approaches when used for accelerating DSE-based EMT simulations.

1.7 Outline of the Thesis

The current chapter presents the introductory material as well as a literature review of the existing techniques for speeding up transient simulations of power systems, and the state-of-the-art partitioning methods for enabling parallel EMT simulations. The rest of the thesis is organized as follows:

Chapter 2

In Chapter 2, an alternative method is investigated for formulating the equations of a system for EMT simulations. It uses Descriptor State-space Equations (DSE) to represent the dynamics of a circuit. A procedure to interface a DSE-based formulation with a Companion Circuits (CC)-based EMT simulator is also developed. This procedure can be used to speed up the simulations using parallel processing.

Chapter 3

In Chapter 3, sparse matrix-based parallelization methods are used for speeding up DSE-based EMT simulations. In particular, two commonly used methods in the literature are adapted and utilized, and their computational performance is compared. The first method transforms a sparse matrix to a Block Diagonal (BD) form and can be used only when the simulated network contains distributed parameter models for transmission lines and/or cables. On the other hand, the second method transforms a sparse matrix to a Bordered Block Diagonal (BBD) form and is more general in its application (i.e., it does not need the presence of any special elements like distributed parameter lines/cables in the simulated system; it only depends on the sparsity structure of the matrix).

Chapter 4

In Chapter 4, a novel delay-based interface for EMT simulations is developed which uses existing inductors in the circuit to partition the network thus allowing for simulation speed up when the solution of the partitioned network is computed on a parallel computing platform.

The proposed interface is guaranteed to be passive thus benefiting the numerical stability of the simulation and also has superior performance compared to existing inductor-based partitioning approaches.

Chapter 5

In Chapter 5, a novel approach for assessing the numerical stability of multi-rate EMT simulations of linear time-invariant (LTI) circuits is developed. It is shown that such simulations always yield a periodically varying system in the discrete time domain. By exploiting this property and applying the well-known 'lifting' technique [56], a sampled data time-invariant representation is obtained for the simulated discrete-time system, which is then used for stability assessment using eigenvalue analysis.

Chapter 6

In Chapter 6, the major conclusions of the thesis are discussed followed by recommendations for future work.

Chapter 2

Co-simulation of DSE-based and CC-based EMT Simulators

This chapter investigates an alternative method for formulating the equations of a network for EMT simulations. This method uses Descriptor State-space Equations (DSE) to represent the dynamical equations of a circuit. A procedure for interfacing a DSE-based formulation with a Companion Circuits (CC)-based EMT simulator is also developed. This procedure enables interfacing of arbitrary power networks with any commercial CC-based EMT simulation package and can also be used to speed up the simulation using parallel processing.

2.1 Introduction

In Chapter 1, a brief introduction to Electromagnetic Transient (EMT) simulations was presented. At its most basic level, every EMT simulator solves the dynamical equations of a circuit using a suitable numerical integration method. Most commercial EMT programs available today use the fixed time-step trapezoidal method [57–59]. It is well-known that mainly there are two approaches to formulate the equations of a given circuit for EMT simulations [8]. The first is the Companion Circuits (CC) approach proposed by Dr. H. W. Dommel [9] which was discussed in Section 1.2. Alternatively, the second approach is to formulate

the state-space equations of the circuit in the form given in (2.1).

$$\underline{\dot{x}}_s = \mathbf{A_s}\underline{x}_s + \mathbf{B_s}\underline{u} \tag{2.1}$$

Here $\underline{x}_s$ is a vector of state variables typically consists of linearly independent inductor currents/flux-linkages and capacitor voltages/charges [8] (the subscript 's' stands for standard state-space form); $\underline{u}$ is the input vector; $\mathbf{A_s}$ is the state matrix while $\mathbf{B_s}$ is the input matrix. Once (2.1) has been formulated, we can use any numerical integration algorithm to compute the updated values of the states $\underline{x}_s$ at every time-step knowing inputs $\underline{u}$.

Traditionally, graph theory-based methods have been used for formulating the state-space equations of a circuit for EMT simulations [8,60]. But these methods require many intermediate matrix manipulation steps which makes them inefficient (w.r.t time and memory) and thus impractical for EMT simulations of large networks [61,62].

In this chapter, we investigate an alternative method to formulate the state-space equations of a circuit. It uses Descriptor State-space Equations (DSE) that are formulated using Modified Nodal Analysis (MNA). Unlike classical state variables which are always linearly independent, descriptor state variables can be linearly dependent. This avoids special considerations for all inductor-current source cutsets or all capacitor-voltage source loops, which are required in many strict state variable formulations. Hence, the DSEs can be automatically formulated for any arbitrary circuit using its netlist (a procedure for doing this is presented later in this chapter).

Once the DSEs of a circuit are formulated, they can be discretized using an implicit integration method (such as the trapezoidal integration method) and used for EMT simulations. This approach is compared with the widely used Companion Circuit (CC) approach. The mathematical equivalence of discretized DSE-based EMT simulations with CC-based EMT simulations has been already proven in [63]. Using various test cases, the advantages and disadvantages of the CC-based and DSE-based approaches are discussed.

The main advantages of DSE-based formulation over classical state-space formulations are: 1) it can be done automatically without any large matrix manipulations, and 2) gives

matrices that are sparse thus making it suitable for large systems. Also, the set of DSEs describing the network immediately allow for the application of widely available linear system analysis tools such as the calculation of system eigenvalues of the real-world network. Although it is not impossible to extract the eigenvalues from CC-based approaches, computing them requires additional post-processing [1] because the state-space equations in continuous time-domain are never explicitly formulated.

Finally, it is shown that a discretized DSE-based formulation can also be easily interfaced with a CC-based EMT simulator without any time-step delay errors. A step-by-step procedure for accomplishing this is derived later in this chapter. This procedure enables interfacing of arbitrary power networks with any commercial CC-based EMT simulation package and can also be used to speed up the simulation using parallel processing.

2.2 Formulation of Equations for EMT Simulations

2.2.1 Companion Circuit (CC)-based EMT Simulations

This section briefly reviews the widely used CC-based EMT simulation [9], so that it can later be contrasted with DSE-based EMT simulation. As discussed in Section 1.2, the difference equations of the circuit in the CC-based approach are formulated using Modified Nodal Analysis (MNA) in the form given in (2.2). Expanding (2.2) gives (2.3).

$$\mathbf{Y}\,\underline{V}(t) = \underline{J}(t) \tag{2.2}$$

$$\begin{bmatrix} \mathbf{G_D} & \mathbf{A_{vs}} \\ \mathbf{A_{vs}^T} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \underline{v}_N(t) \\ \underline{i}_S(t) \end{bmatrix} = \begin{bmatrix} \underline{i}_{HS}(t) \\ \underline{v}_S(t) \end{bmatrix} \tag{2.3}$$

Here,

$\mathbf{G_D}$: Admittance matrix of the discretized circuit.

$\mathbf{A_{vs}}$: Incidence matrix for independent voltage source branches.

$\underline{v}_N(t)$: Node voltages at time t.

$\underline{i}_S(t)$: Currents through independent voltage sources at time t.

$\underline{i}_{HS}(t)$: History current sources, and independent current sources at time t.

$\underline{v}_S(t)$: Independent voltage sources at time t.

Any CC-based EMT simulator uses the following general procedure:

1. Form the $\mathbf{Y}$ matrix using the circuit's netlist.
2. Initialize $\underline{V}(t)$.
3. Advance the solution time $t = t + \Delta t$.
4. Update $\underline{J}(t)$ by computing the history current values for each inductor and capacitor, and the values of independent voltage and current sources.
5. If switches are present, check if any switch has changed its state and modify the G matrix accordingly.
6. Compute $\underline{V}(t)$ by solving (2.2).
7. Update the currents $i_L(t)$ and $i_C(t)$ for each inductor and capacitor using the history current values from Step 4 and $\underline{V}(t)$ from Step 5.
8. Go back to Step 3 if end time not reached.

The case of other non-linearities such as inductor saturation can also be accommodated by modelling them with parallel inductors with switches [64], [65]. As we can see, the CC-based EMT simulation approach is highly scalable which is why it is used in the majority of commercial EMT simulation packages today [57], [58], [59]. The next section reviews the procedure for formulating $\mathbf{Y}$ using a circuit's netlist.

Formulation of the Y Matrix

We know from (2.3) that $\mathbf{Y}$ is composed of sub-matrices $\mathbf{G_D}$ and $\mathbf{A_{vs}}$. The sizes of these sub-matrices are given in Table 2.1. Here, n_n is the number of nodes, and n_{vs} is the number of independent voltage sources in the circuit.

Table 2.1: Size of sub-matrices in Y

Matrix	Size
$\mathbf{G_D}$	$n_n \times n_n$
$\mathbf{A_{vs}}$	$n_n \times n_{vs}$

The matrix $\mathbf{G_D}$ includes all the conductance corresponding to resistors as well as discretized inductors and capacitors present in the circuit. The general expression for (i, j) entry of $\mathbf{G_D}$ is given by (2.4).

$$\mathbf{G_D}(i,j) = \begin{cases} g_i + \sum\limits_{\substack{k=1 \\ k \neq i}}^{n_n} g_{ik}, & \text{if } i = j \\ -g_{ij}, & \text{if } i \neq j \end{cases} \tag{2.4}$$

Here g_i is the sum of all the conductance between node i and ground while g_{ij} is the conductance between node i and j where both i and j vary from 1 to n_n.

The matrix $\mathbf{A_{vs}}$ is the incidence matrix corresponding to independent voltage source branches. One of the ways of formulating it is using (2.5).

$$\mathbf{A}(i,k) = \begin{cases} 1, & \text{if branch } k \text{ has start node } i \\ -1, & \text{if branch } k \text{ has end node } i \\ 0, & \text{otherwise} \end{cases} \tag{2.5}$$

2.2.2 Descriptor State-space Equation (DSE)-based EMT Simulations

For DSE-based EMT simulations, the equations of the circuit are formulated firstly in the continuous time-domain using MNA in the form given in (2.6)[1].

$$\mathbf{E}\dot{\underline{x}} = -\mathbf{A}\underline{x} + \mathbf{B}\underline{u} \tag{2.6}$$

[1]Note that the matrix $\mathbf{E}$ is often singular, so simply inverting it to obtain the classical state variable form (of (2.1)) is generally not possible [66]

Once (2.6) has been formulated, it is discretized using the trapezoidal integration method with a time-step Δt. This gives the update equation as in (2.7) which can then be used to carry out EMT simulations studies.

$$\left(\mathbf{E} + \frac{\mathbf{A}\Delta t}{2}\right)\underline{x}(t) = \left(\mathbf{E} - \frac{\mathbf{A}\Delta t}{2}\right)\underline{x}(t-\Delta t) + \frac{\mathbf{B}\Delta t}{2}\left(\underline{u}(t) + \underline{u}(t-\Delta t)\right) \tag{2.7}$$

Equation (2.6) is called the Descriptor State-space Equation (DSE) where $\underline{x}$ is the vector of descriptor state variables (NOT the same as $\underline{x}_s$ in (2.1)) and $\underline{u}$ is the input vector, both as given in (2.8).

$$\underline{x} = \begin{bmatrix}\underline{v}_N^T & \underline{i}_L^T & \underline{i}_S^T\end{bmatrix}^T \qquad \underline{u} = \begin{bmatrix}\underline{v}_S^T & \underline{j}_S^T\end{bmatrix}^T \tag{2.8}$$

Here,

$\underline{v}_N$: Node voltages.

$\underline{i}_L$: Inductor currents.

$\underline{i}_S$: Currents through independent voltage sources.

$\underline{v}_S$: Independent voltage sources.

$\underline{j}_S$: Independent current sources.

When (2.6) is expanded using $\underline{x}$ and $\underline{u}$ from (2.8), we get (2.9).

$$\begin{bmatrix}\mathbf{C} & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{L} & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{0}\end{bmatrix}\frac{d}{dt}\begin{bmatrix}\underline{v}_N\\ \underline{i}_L\\ \underline{i}_S\end{bmatrix} = -\begin{bmatrix}\mathbf{G} & \mathbf{A_L} & \mathbf{A_{vs}}\\ \mathbf{A_L^T} & \mathbf{0} & \mathbf{0}\\ \mathbf{A_{vs}^T} & \mathbf{0} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\underline{v}_N\\ \underline{i}_L\\ \underline{i}_S\end{bmatrix} + \begin{bmatrix}\mathbf{0} & \mathbf{A_{js}}\\ \mathbf{0} & \mathbf{0}\\ -\mathbf{I} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\underline{v}_S\\ \underline{j}_S\end{bmatrix} \tag{2.9}$$

Here,

$\mathbf{C}$: Capacitance matrix.

$\mathbf{L}$: Diagonal matrix containing inductance values.

$\mathbf{G}$: Conductance matrix (corresponding to lumped resistors in the circuit)

$\mathbf{A_L}$: Incidence matrix for inductor branches.

$\mathbf{A_{vs}}$: Incidence matrix for voltage source branches.

$\mathbf{A_{js}}$: Incidence matrix for current source branches.

In most cases, the matrix **E** is singular as seen in (2.9). The exception is the relatively rare case when there are no voltage sources (i.e., the last block row of **E** in (2.9) does not exist) and **C** is also non-singular. Singularity of **E** implies that the entries of the descriptor state vector $\underline{x}$ are linearly dependent. Also, when **E** is singular, an explicit integration method like Forward Euler cannot be used for numerical integration [66].

A detailed derivation of (2.9) can be found in [63]. If the sub-equations in (2.9) are expanded, we get (2.10) – (2.12).

$$\mathbf{C}\,\frac{d\underline{v}_N}{dt} = -\mathbf{G}\,\underline{v}_N - \mathbf{A_L}\,\underline{i}_L - \mathbf{A_{vs}}\,\underline{i}_S - \mathbf{A_{js}}\,\underline{j}_S \tag{2.10}$$

$$\mathbf{L}\,\frac{d\underline{i}_L}{dt} = \mathbf{A_L^T}\,\underline{v}_N \tag{2.11}$$

$$0 = \mathbf{A_{vs}^T}\,\underline{v}_N - \underline{v}_S \tag{2.12}$$

Taking a closer look at (2.10) – (2.12), we can conclude the following:

- Equation (2.10) is obtained by applying Kirchhoff's Current Law (KCL) at every node.
- Equation (2.11) is obtained by applying Kirchhoff's Voltage Law (KVL) in every inductor branch.
- Equation (2.12) is obtained by applying KVL in every independent voltage source branch.

Based on the above discussion, we can say that a DSE-based EMT simulator can use the following general procedure:

1. Form the **E**, **A** and **B** matrices using the circuit's netlist.
2. Initialize $\underline{x}(t)$ and $\underline{u}(t)$.
3. Advance the solution time $t = t + \Delta t$.
4. Update $\underline{u}(t)$ by computing the independent voltage source and current source values at time t.

5. Compute the RHS of (2.7) from the history values and $\underline{u}(t)$ from Step 4.

6. If switches are present, check if any switch has changed its state and modify the **A** matrix accordingly.

7. Compute $\underline{x}(t)$ by solving (2.7) using the RHS computed in Step 5.

8. Go back to Step 3 if end time not reached.

From the above discussion, we can see that this approach is also highly scalable (just like the CC-based approach). The key step is the automatic generation of **E**, **A** and **B** (from (2.6)) for any arbitrary circuit using its netlist. This is summarized below.

Formulation of the E, A and B Matrices

In this section, we present a procedure to automatically formulate **E**, **A** and **B**. We know from (2.9) that each of these are composed of other sub-matrices (**C**, **G**, **L** etc.). The sizes of the different sub-matrices are given in Table 2.2. Here, n_n is the number of nodes, n_l is the number of inductors, n_{vs} is the number of independent voltage sources, and n_{js} is the number of independent current sources in the circuit.

Table 2.2: Size of sub-matrices in E, A and B

Matrix	Size
C & **G**	$n_n \times n_n$
L	$n_l \times n_l$
$\mathbf{A_L}$	$n_n \times n_l$
$\mathbf{A_{vs}}$	$n_n \times n_{vs}$
$\mathbf{A_{js}}$	$n_n \times n_{js}$

The matrix **C** includes all the capacitors present in the circuit. The general expression for (i, j) entry of **C** is given by (2.13), where c_i is the sum of all the capacitance between node i and ground while c_{ij} is the capacitance between node i and j. Here both i and j vary

from 1 to n_n.

$$\mathbf{C}(i,j) = \begin{cases} c_i + \sum\limits_{\substack{k=1 \\ k\neq i}}^{n_n} c_{ik}, & \text{if } i = j \\ -c_{ij}, & \text{if } i \neq j \end{cases} \tag{2.13}$$

The matrix $\mathbf{G}$ includes only the entries corresponding to lumped resistors present in the circuit. The general expression for (i,j) entry of $\mathbf{G}$ is given by (2.14), where g_i is the sum of all the conductance between node i and ground while g_{ij} is the conductance between node i and j. Here both i and j vary from 1 to n_n.

$$\mathbf{G}(i,j) = \begin{cases} g_i + \sum\limits_{\substack{k=1 \\ k\neq i}}^{n_n} g_{ik}, & \text{if } i = j \\ -g_{ij}, & \text{if } i \neq j \end{cases} \tag{2.14}$$

The matrix $\mathbf{L}$ is a diagonal matrix which includes all the inductors present in the circuit. The general expression for (k,k) entry of $\mathbf{L}$ is given by (2.15) where l_k is the k^{th} inductance in the circuit's netlist. Here k varies from 1 to n_l.

$$\mathbf{L}(k,k) = l_k \tag{2.15}$$

The matrices $\mathbf{A_L}$, $\mathbf{A_{vs}}$ and $\mathbf{A_{js}}$ are incidence matrices corresponding to inductor branches, independent voltage source branches and independent current source branches respectively. Each of these can be formulated using (2.5).

2.3 Comparison of CC-based and DSE-based EMT Simulations

2.3.1 Comparison of Simulations Results

Firstly, we compare the simulation results from the two approaches. For this, consider a simple power system example as shown in Fig. 2.1. Transmission lines are modelled by

coupled π-sections and loads as three-phase resistors, inductors, and capacitors. System data is given in Table 2.3.

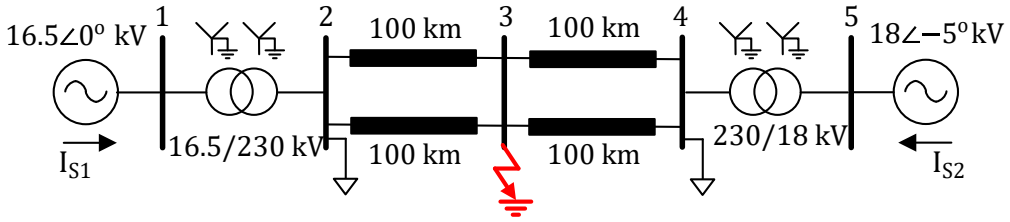
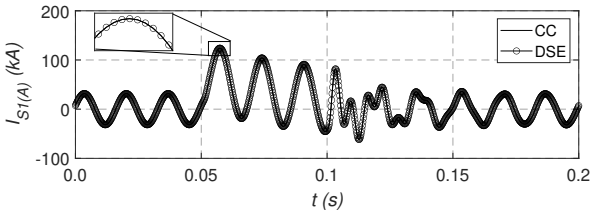


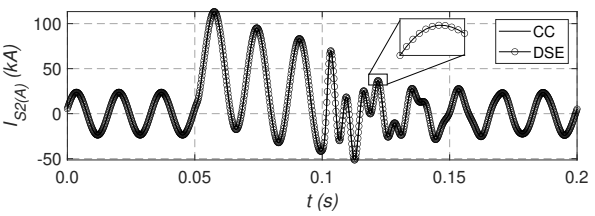
Figure 2.1: A Simple Power System Example

Table 2.3: Data for the system of Fig. 2.1

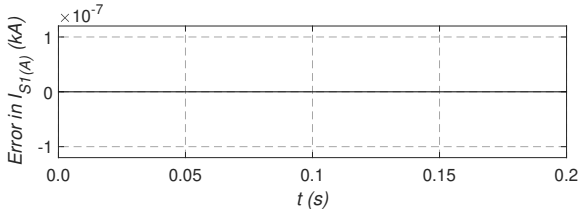
Transmission Line
1000 MVA, 230 kV
$R_+ = 1.05 \times 10^{-5}\ pu/m$; $X_+ = 1.233 \times 10^{-4}\ pu/m$; $B_+ = 1.6987 \times 10^{-5}\ pu/m$
$R_0 = 1.0658 \times 10^{-4}\ pu/m$; $X_0 = 3.211 \times 10^{-4}\ pu/m$; $B_0 = 1.2064 \times 10^{-5}\ pu/m$
Transformers
1: 1000 MVA, 16.5/230 kV, $X_l = 0.181\ pu$, $I_{mag} = 2\%$
2: 1000 MVA, 230/18 kV, $X_l = 0.181\ pu$, $I_{mag} = 2\%$
Loads
522 MW, 150 MVAr



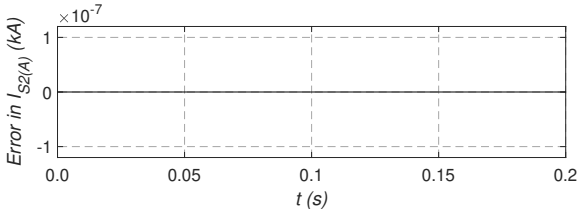
(a) Phase A current of Source 1 ($\mathbf{I_{S1(A)}}$)



(b) Phase A current of Source 2 ($\mathbf{I_{S2(A)}}$)



(c) Absolute Error in $\mathbf{I_{S1(A)}}$



(d) Absolute Error in $\mathbf{I_{S2(A)}}$

Figure 2.2: Simulation Results for the system of Fig. 2.1

A time-step of 200 μs is used for both the approaches. The system initially operates in steady state. At $t = 0.05\ s$, a 50 ms (three cycles) solid three phase fault occurs at Bus 3. A comparison of the simulation results from the two approaches is shown in Fig. 2.2.

We can see that the simulation results from both the approaches are essentially exactly identical. The maximum absolute error in both $I_{S1(A)}$ and $I_{S2(A)}$ is zero. This must be so since these two approaches are mathematically equivalent [63].

2.3.2 Computation of Eigenvalues

In addition to providing a framework for EMT simulation, the DSE formulation has an additional advantage which is lacking in traditional CC-based EMT solvers. It is that it directly permits the application of analytical tools for linear systems (e.g., eigenvalue analysis, root locus etc.). Although classical state-space (SS) formulations can also be used for this, formulating equations in classical SS form may involve many intermediate matrix manipulation steps. This makes them inefficient (w.r.t time and memory) and thus impractical for formulating the equations of large systems [61, 62].

An appropriately simple example as in Fig. 2.3 is chosen to demonstrate the basic principles. The example includes an all capacitor-voltage source (C-V) loop $v_S - v_{C2} - v_{C3} - v_{C1}$ which highlights the advantage of DSE-based formulation over classical SS formulation.

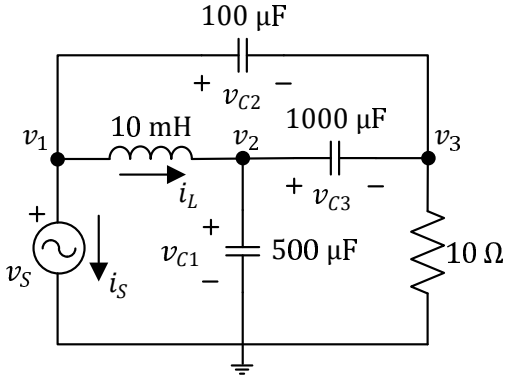


Figure 2.3: Simple RLC Circuit (Computation of Eigenvalues)

For the DSEs, $\underline{x} = [v_1 \quad v_2 \quad v_3 \quad i_L \quad i_S]^T$, and then the **E** and **A** matrices are as in (2.16)

and (2.17).

$$\mathbf{E} = \begin{bmatrix} 0.0001 & 0 & -0.0001 & 0 & 0 \\ 0 & 0.0015 & -0.001 & 0 & 0 \\ -0.0001 & -0.001 & 0.0011 & 0 & 0 \\ 0 & 0 & 0 & 0.01 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.16}$$

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0.1 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.17}$$

Due to the C-V loop $v_S - v_{C2} - v_{C3} - v_{C1}$, only two out of the three capacitor voltages can be state variables in a classical SS formulation. Hence, $\underline{x} = [v_{C1} \quad v_{C2} \quad i_L]^T$ is taken for formulating the equations in classical SS form (as in (2.1)). Note, to eliminate the extra algebraic variable v_{C3} in this case, intermediate matrix manipulations are needed. After eliminating v_{C3}, we finally get the state matrix $\mathbf{A_s}$ as given in (2.18).

$$\mathbf{A_s} = \begin{bmatrix} 0 & 153.85 & 1692.31 \\ 0 & -230.77 & -1538.46 \\ -100 & 0 & 0 \end{bmatrix} \tag{2.18}$$

To corroborate the fact that one can obtain the system eigenvalues from a DSE-based formulation, a comparison of the eigenvalues of $\mathbf{A_s}$ with the eigenvalues of the matrix pencil $(-\mathbf{A}, \mathbf{E})$ is given in Table 2.4. In Table 2.4, $\sigma(\mathbf{A_s})$ are the system eigenvalues. We can clearly see that $\sigma(\mathbf{A_s}) \subseteq \sigma(-\mathbf{A}, \mathbf{E})$. In fact, the two extra $-\infty$ eigenvalues arise in the case of $(-\mathbf{A}, \mathbf{E})$ because $\mathbf{E}$ here is rank-deficient by two [49], which implies that only three elements of the descriptor state vector $\underline{x}$ (out of five) are linearly independent.

It is worthwhile to note the ease with which C-V loops are handled in a DSE-based

Table 2.4: Comparison of Eigenvalues

Eigenvalues of $\mathbf{A_s}$ $[\sigma(\mathbf{A_s})]$	**Eigenvalues of $(-\mathbf{A}, \mathbf{E})$** $[\sigma(-\mathbf{A}, \mathbf{E})]$
$-66.139 \pm j389.65$	$-66.139 \pm j389.65$
-98.491	-98.491
	$-\infty$, $-\infty$

formulation. Unlike a classical SS formulation, there is no requirement in a DSE-based formulation to eliminate the intermediate algebraic variables. The same can be concluded if all inductor-current source (L-J) cutsets are present in a circuit.

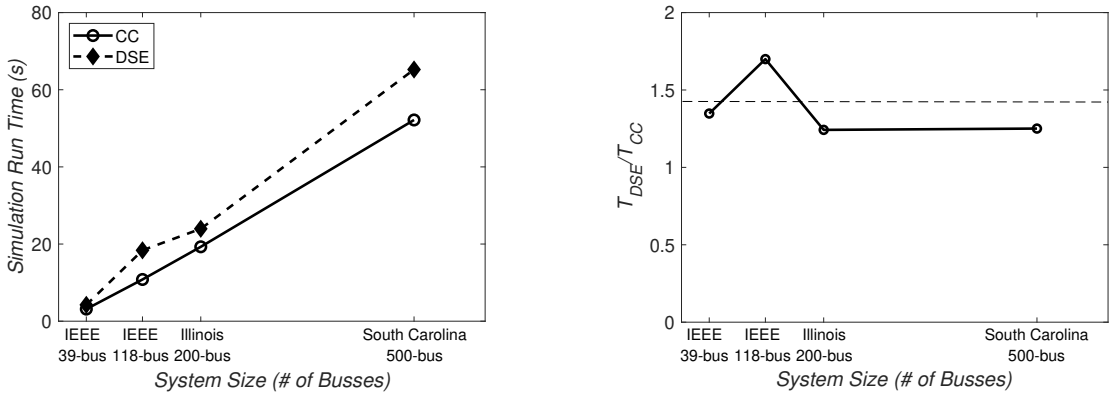
Note that although in the above discussion we have demonstrated the computation of eigenvalues with DSE-based formulation using a relatively simple example, it can be easily accomplished for any arbitrary circuit using the procedure to automatically formulate **A** and **E** given earlier in Section 2.2.2.

2.3.3 Comparison of CPU Run Times

To investigate how the computer run-times for DSE-based and CC-based simulation approaches scale with increasing system sizes, we consider four systems. These include standard IEEE test systems (39-bus, and 118-bus) as well as synthetic versions [67] of the Illinois 200-bus system and the South Carolina 500-bus system. The data for the latter two is available in the public domain [68]. In each case, a 10 s simulation is performed with a time-step of 50 μs. Transmission lines in each of these systems have been modelled by coupled π-sections and loads as three-phase resistors, inductors, and capacitors.

Fig. 2.4a gives a comparison of the CPU run times for the CC-based and DSE-based approaches as a function of the number of three phase busses in the system. As shown in Fig. 2.4b, on an average the CC-based approach is about 1.4 times faster for the cases considered here. Hence, we can say that even though CC-based and DSE-based approaches are mathematically equivalent (as shown in [63]), the DSE-based approach has a higher CPU run time. Note however that although the DSE-based simulation approach is slower,

it directly yields additional information such as system eigenvalues, which the CC-based approach does not.



(a) CPU Run Times **(b) Ratio of CPU Run Times**

Figure 2.4: Comparison of CPU Run Times

To understand why this difference exists, consider the size of the matrices involved in the computations in every time-step for each of the approaches. The $\mathbf{Y}$ matrix (in (2.2)) for the CC-based approach is a square matrix of size $(n_n + n_{vs})$. On the other hand, the $\left(\mathbf{E} + \frac{\mathbf{A}\Delta t}{2}\right)$ matrix (in (2.7)) for the DSE-based approach is a square matrix of size $(n_n + n_l + n_{vs})$. Here, n_n is the number of nodes, n_{vs} is the number of independent voltage sources and n_l is the number of inductors. Hence, we can conclude that the superior speed of the CC-based approach can be attributed to the fact that the size of the matrix involved in its computations at every time-step is smaller than that in a DSE-based approach.

2.4 Interfacing of a DSE-based Formulation with a CC-based EMT Simulator

We will now present a procedure for interfacing a DSE-based formulation with a CC-based EMT simulator. This procedure is similar to the ones presented in [42, 43] but derived here for a DSE-based formulation. Consider the network to be interfaced is represented by a black box as shown in Fig. 2.5 with n ports for interfacing.

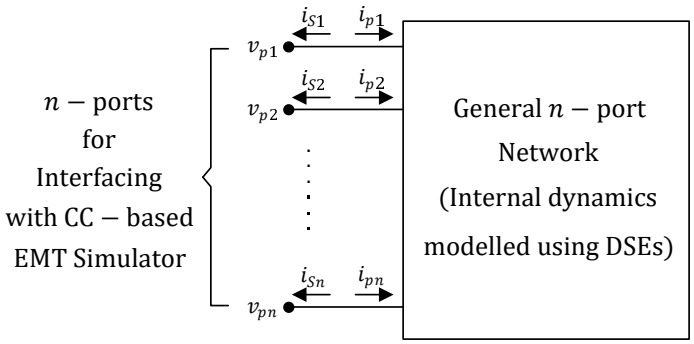


Figure 2.5: A General n-Port Network

Consider that this network has lumped circuit elements (viz. resistors (R), inductors (L) and capacitors (C)) and internal voltage and/or current sources (each of which may be dependent or independent). Then, internal dynamics of this network can be easily modelled in continuous time-domain using DSEs. For doing this, let $\underline{v}_p$, $\underline{i}_S$ and $\underline{i}_p$ be as given in (2.19), (2.20) and (2.21) respectively.

$$\underline{v}_p = \begin{bmatrix} v_{p1} & v_{p2} & \dots & v_{pn} \end{bmatrix}^T \tag{2.19}$$

$$\underline{i}_S = \begin{bmatrix} i_{S1} & i_{S2} & \dots & i_{Sn} \end{bmatrix}^T \tag{2.20}$$

$$\underline{i}_p = \begin{bmatrix} i_{p1} & i_{p2} & \dots & i_{pn} \end{bmatrix}^T \tag{2.21}$$

Then, the DSEs for this network in continuous time-domain can be written in the form given in (2.22).

$$\mathbf{E}\underline{\dot{x}} = -\mathbf{A}\underline{x} + \mathbf{B}\underline{v}_p + \mathbf{B_i}\underline{u}_i \tag{2.22}$$

Here, $\underline{x} = \begin{bmatrix} \underline{v}_N^T & \underline{i}_L^T & \underline{i}_{vs}^T & \underline{i}_S^T \end{bmatrix}^T$; $\underline{v}_N$ is the vector of internal node voltages, $\underline{i}_L$ is the vector of internal inductor currents, $\underline{i}_{vs}$ is the vector of branch currents through internal voltage sources, and $\underline{u}_i$ is the vector of internal sources. In addition to this, it can be easily verified that the relationship between $\underline{i}_p$ and $\underline{x}$ is as given in (2.23).

$$\underline{i}_p = \mathbf{B}^T\underline{x} \tag{2.23}$$

Now, to interface this n-port network with a CC-based EMT simulator, we need a discrete

time-domain relationship between $\underline{i}_p(t)$ and $\underline{v}_p(t)$ as given by (2.24).

$$\underline{i}_p(t) = \mathbf{G_n}\underline{v}_p(t) + \underline{I}_{HIST}(t-\Delta t) \tag{2.24}$$

This relationship can be derived from the DSE-based formulation by firstly discretizing (2.22) using the trapezoidal integration method to give (2.25).

$$\begin{aligned}\underline{x}(t) = \left(\mathbf{E} + \frac{\mathbf{A}\Delta t}{2}\right)^{-1} \Bigg\{ &\left(\mathbf{E} - \frac{\mathbf{A}\Delta t}{2}\right)\underline{x}(t-\Delta t) + \frac{\mathbf{B}\Delta t}{2}\left(\underline{v}_p(t) + \underline{v}_p(t-\Delta t)\right) \\ &+ \frac{\mathbf{B_i}\Delta t}{2}\left(\underline{u}_i(t) + \underline{u}_i(t-\Delta t)\right)\Bigg\}\end{aligned} \tag{2.25}$$

Subsequently, substituting (2.25) in (2.23) gives $\underline{i}_p(t)$ as in (2.26).

$$\begin{aligned}\underline{i}_p(t) = {} & \mathbf{B}^T\left(\mathbf{E} + \frac{\mathbf{A}\Delta t}{2}\right)^{-1}\frac{\mathbf{B}\Delta t}{2}\underline{v}_p(t) \\ & + \mathbf{B}^T\left(\mathbf{E} + \frac{\mathbf{A}\Delta t}{2}\right)^{-1}\left(\mathbf{E} - \frac{\mathbf{A}\Delta t}{2}\right)\underline{x}(t-\Delta t) \\ & + \mathbf{B}^T\left(\mathbf{E} + \frac{\mathbf{A}\Delta t}{2}\right)^{-1}\frac{\mathbf{B}\Delta t}{2}\underline{v}_p(t-\Delta t) \\ & + \mathbf{B}^T\left(\mathbf{E} + \frac{\mathbf{A}\Delta t}{2}\right)^{-1}\frac{\mathbf{B_i}\Delta t}{2}\left(\underline{u}_i(t) + \underline{u}_i(t-\Delta t)\right)\end{aligned} \tag{2.26}$$

We can see that (2.26) has the same form as (2.24) with $\mathbf{G_n}$ and $\underline{I}_{HIST}(t-\Delta t)$ as given in (2.27) and (2.28) respectively.

$$\mathbf{G_n} = \mathbf{B}^T\left(\mathbf{E} + \frac{\mathbf{A}\Delta t}{2}\right)^{-1}\frac{\mathbf{B}\Delta t}{2} \tag{2.27}$$

$$\begin{aligned}\underline{I}_{HIST}(t-\Delta t) = {} & \mathbf{B}^T\left(\mathbf{E} + \frac{\mathbf{A}\Delta t}{2}\right)^{-1}\left(\mathbf{E} - \frac{\mathbf{A}\Delta t}{2}\right)\underline{x}(t-\Delta t) \\ & + \mathbf{B}^T\left(\mathbf{E} + \frac{\mathbf{A}\Delta t}{2}\right)^{-1}\frac{\mathbf{B_i}\Delta t}{2}\left(\underline{u}_i(t) + \underline{u}_i(t-\Delta t)\right) \\ & + \mathbf{B}^T\left(\mathbf{E} + \frac{\mathbf{A}\Delta t}{2}\right)^{-1}\frac{\mathbf{B}\Delta t}{2}\underline{v}_p(t-\Delta t)\end{aligned} \tag{2.28}$$

Thus, we have the required discrete time-domain relationship between $\underline{i}_p(t)$ and $\underline{v}_p(t)$. Note, since $\underline{u}_i$ is a vector of internal sources, it is already known for time t and hence can be part

of $\underline{I}_{HIST}(t-\Delta t)$.

Now we can interface this network with any CC-based EMT simulator using the following procedure:

1. Form **E**, **A** and **B** matrices for the network to be interfaced using the procedure given in Section 2.2.2 and then calculate the Norton equivalent admittance ($\mathbf{G_n}$) using (2.27).
2. Add $\mathbf{G_n}$ to the overall **Y** matrix of the CC-based EMT simulator.
3. Initialize $\underline{x}(t)$.
4. Advance the solution time $t = t + \Delta t$.
5. Knowing $\underline{x}(t-\Delta t)$ and $\underline{v}_p(t-\Delta t)$, calculate $\underline{I}_{HIST}(t-\Delta t)$ using (2.28). Then, add $\underline{I}_{HIST}(t-\Delta t)$ to the overall $\underline{J}(t)$ of the CC-based simulator (see (2.2)).
6. If switches are present inside the interfaced network, check if any switch has changed its state and modify $\mathbf{G_n}$ accordingly.
7. Update the $\mathbf{G_n}$ added to the overall **Y** matrix of the CC-based EMT simulator.
8. The CC-based simulator computes $\underline{V}(t)$ by solving (2.2).
9. Read $\underline{v}_p(t)$ from the overall $\underline{V}(t)$ of the CC-based EMT simulator (see (2.2)). Then, calculate $\underline{x}(t)$ using (2.25).
10. Go back to Step 4 if end time not reached.

Note that there is no time-step delay involved while interfacing the DSE-based formulation, and the interfaced modelling is mathematically exactly equivalent to a full CC-based approach being used directly [63]. If distributed parameter elements (such as cables or transmission lines) are present in the system, then these can be easily included as CC-models and interfaced with the DSE-based formulation using the given procedure[2].

[2]Distributed parameter elements such as cables or transmission lines can also be included in the DSE-based formulation as shown in Appendix B.

Significant parts of the above algorithm can be inherently computed in parallel, thus making it suitable for implementation on a parallel computing platform. For example, if a large network is divided into multiple subnetworks $\{1, 2, \ ... \ N\}$, then we can see that Steps 5 to 7, and Step 9 for a subnetwork i are completely independent of those for subnetwork j. Therefore, these step sequences in the above procedure can be performed in parallel for each subnetwork. Note that this parallel implementation does not rely on transportation delays introduced by distributed parameter elements like transmissions lines [37].

Also note that Steps 6 and 7 in the above algorithm allow inclusion of time-varying network components (like switches) in DSE-based formulations. Hence it could also be used for modelling complex switching systems like HVdc converters. However, efficient implementation of such systems requires additional steps such as optimal ordering of nodes at which switches are connected to ensure efficient matrix refactorizations [69, 70]. Moreover, traditional CC-based EMT platforms have evolved (efficient) models for various types of HVdc converters and their associated controllers. The ability to interface DSE-based formulation with traditional CC-based EMT (host) platforms is a powerful feature that enables the immediate utilization of the capabilities of the host platform.

Hence, while verifying the above interfacing procedure with examples in the next section, periodically switching topologies (like HVdc converters) are modelled in the CC-based EMT simulator itself while the rest of the network is modelled using a DSE-based formulation interfaced with the CC-based EMT simulator.

2.5 Simulations Examples

To verify the proposed interfacing procedure, we use two standard test cases. The interfacing procedure has been implemented as a subroutine in PSCAD/EMTDC (a commercial CC-based EMT simulator) [71]. As a reference, the entire system is also modelled in the CC-based EMT simulator to compare the accuracy of the interfacing procedure.

2.5.1 Example 1 - IEEE 9-bus System

The first example is the standard IEEE 9-bus system shown in Fig. 2.6. The main objectives of this test case are: (i) to verify the accuracy of the proposed interfacing procedure, (ii) to demonstrate that it can be used to interface arbitrary power networks with a commercial CC-based EMT simulator.

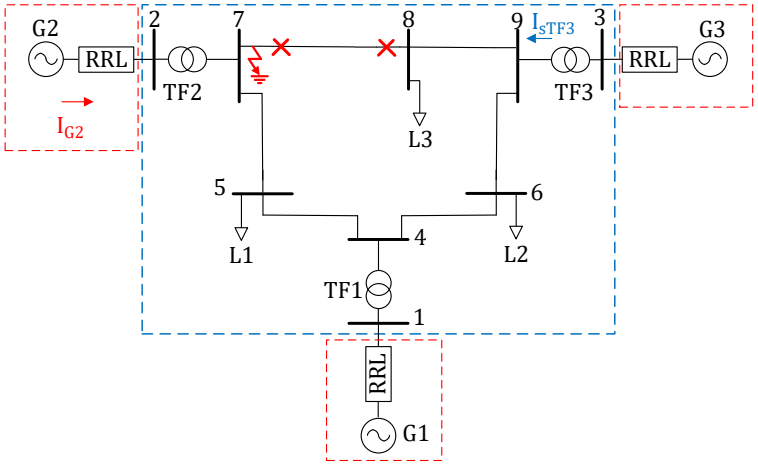
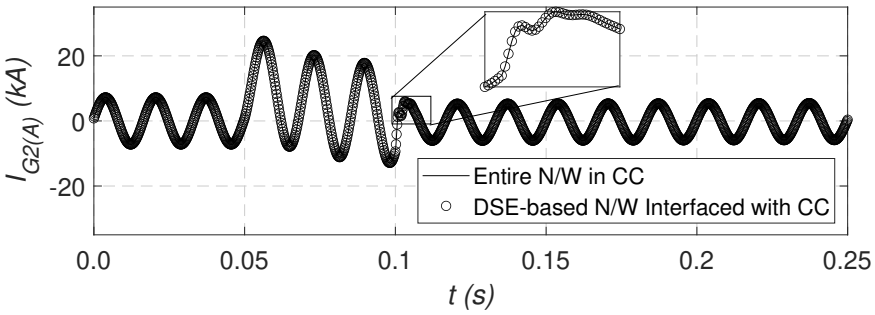


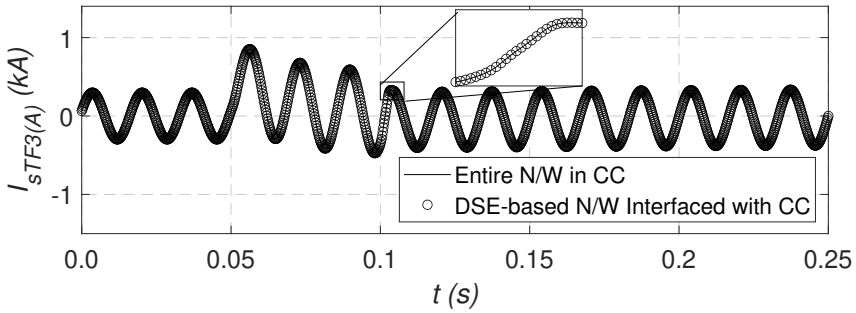
Figure 2.6: Interfacing Example 1 - IEEE 9-bus System

The data for this system has been taken from [72]. All components inside the blue dashed boxes in Fig. 2.6 are modelled using the DSE-based approach and then interfaced, while the components in the red dashed boxes are directly modelled in the commercial CC-based EMT simulator. Transmission lines have been modelled by coupled π-sections, and loads as three-phase resistors, inductors, and capacitors.

A time-step of 200 μs has been used for simulation. The system initially operates in steady state. At t = 0.05 s, a solid three-phase fault occurs near Bus 7. The transmission line 7–8 is opened after three cycles (50 ms) to clear the fault. Note that this causes a change in the Norton equivalent admittance (i.e. $\mathbf{G_n}$ in (2.27)). A comparison of the simulation results is given in Fig. 2.7.



(a) Phase A Current of Source G2 (External Variable)



(b) Bus side Phase A Current of Transformer TF3 (Internal Variable)

Figure 2.7: Simulation Results - Interfacing Example 1 (IEEE 9-bus)

As we can see in Fig. 2.7a and 2.7b, the results for both cases are essentially exactly identical.

2.5.2 Example 2 - IEEE 39-bus System with LCC-HVdc

The second example is the standard IEEE 39-bus system shown in Fig. 2.8. The main objectives of this test case are: (i) to verify the accuracy of the proposed interfacing procedure using a relatively larger system, and (ii) to run the partitioned system on a parallel computing platform to measure its speedup performance.

The data for this system has been taken from [72]. The standard case has been modified to include an embedded LCC-HVdc link in place of the transmission line between Bus 4 and Bus 14. The electrical parameters of the LCC-HVdc link are given in Appendix A. The controls for the dc link are the same as in [73].

The system is partitioned into two parts as shown in Fig. 2.8. All components inside the

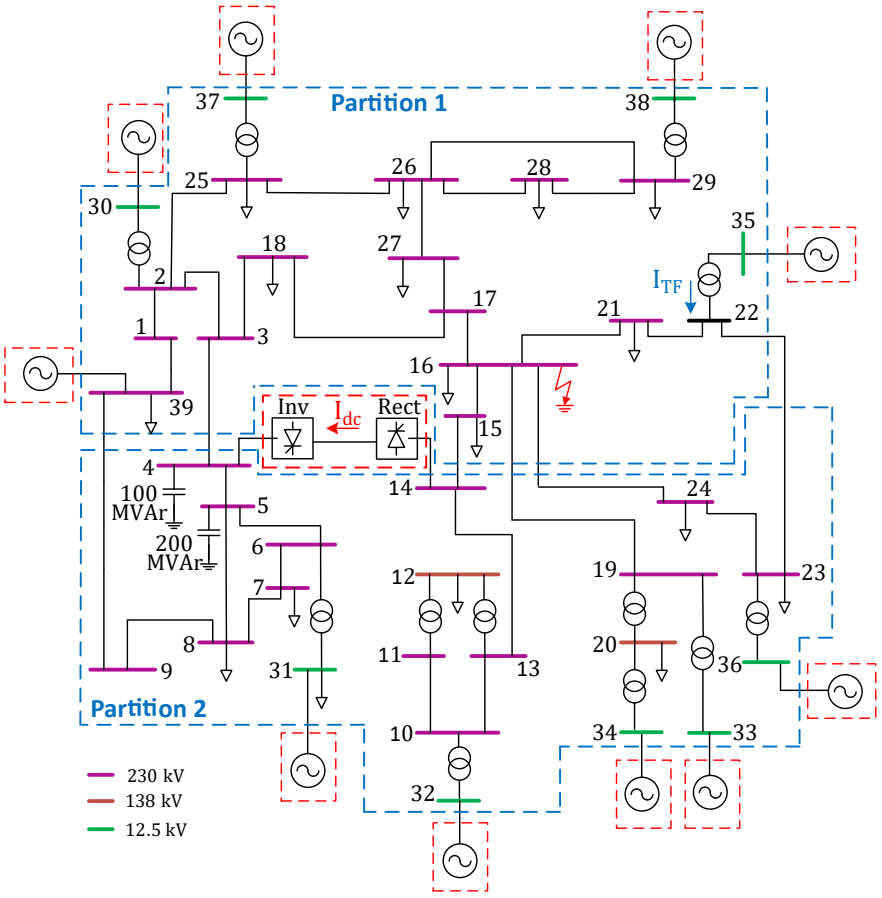


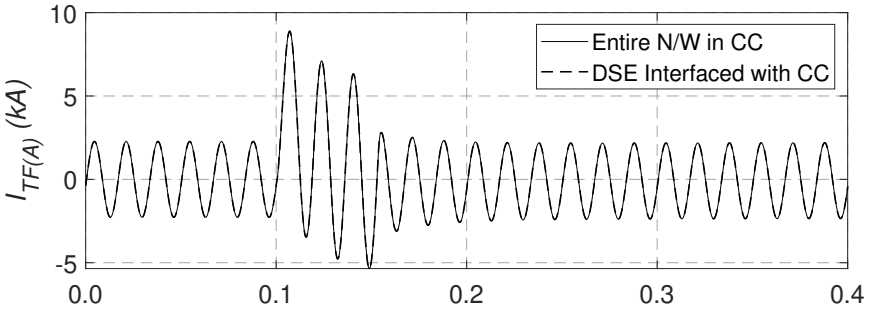
Figure 2.8: Interfacing Example 2 - IEEE 39-bus System with LCC-HVdc

blue dashed boxes are modelled using the DSE-based approach and then interfaced, while the components in the red dashed boxes are directly modelled in the commercial CC-based EMT simulator. Transmission lines are modelled by coupled π-sections, and loads as three-phase resistors, inductors, and capacitors.

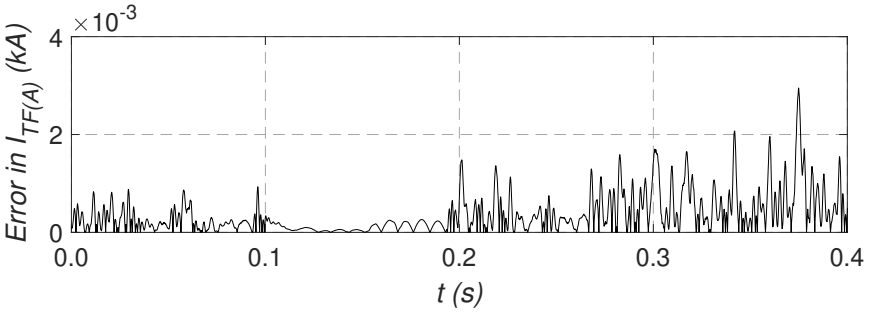
A time-step of 20 μs is used for simulation. The system initially operates in steady state. At t = 0.1 s, a three cycle (50 ms) phase-A-to-ground fault occurs near Bus 16 (internal to Partition #1). Note that this causes a change in the Norton equivalent admittance (i.e. $\mathbf{G_n}$ in (2.27)) of Partition #1.

Fig. 2.9a shows the current in phase A ($I_{TF(A)}$, a variable internal to the DSE-based formulation) flowing into the Bus 22 computed using the full CC-based simulation and the proposed DSE-based formulation interfaced with CC-based EMT simulator while Fig. 2.10a

shows the comparison of the results for the dc link current (I_{dc}, a variable external to the DSE-based formulation) from both the approaches. As we can see, the results match very closely. The maximum absolute error in $I_{TF(A)}$ is 0.003 kA (as shown in Fig. 2.9b) which corresponds to a relative error of 0.15% (relative to the steady state peak value of $I_{TF(A)}$). Similarly, the results for I_{dc} also match very closely (the absolute error in I_{dc} is as shown in Fig. 2.10b).



(a) Phase A Current of Transformer 22-35 on Bus 22 Side (Internal Variable)

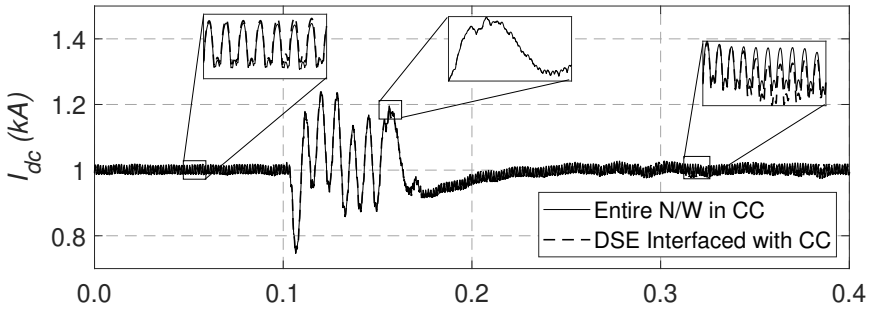


(b) Absolute Error in Phase A Current of Transformer 22-35

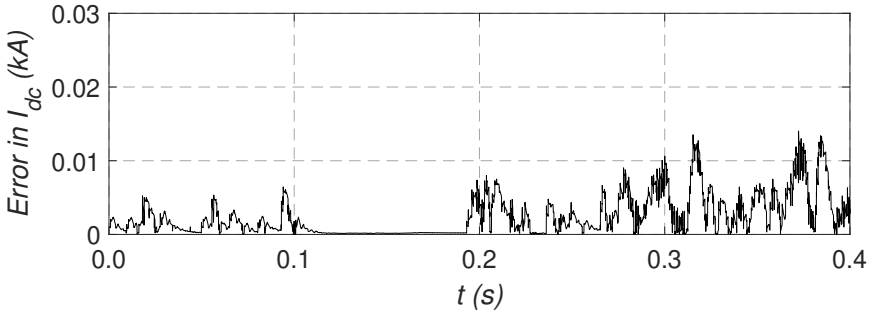
Figure 2.9: Simulation Results for I_{TF} - Interfacing Example 2 (IEEE 39-bus)

Note, the error in this case is not exactly zero (unlike the previous cases) because of the following reason: When simulating continuously switching circuits (like LCC-HVdc), PSCAD/EMTDC, being a fixed time-step simulation program, uses specialized techniques for accurately simulating the switching behaviour. These techniques include interpolation for locating the switching instant [74] and for chatter removal [75] as well as the instantaneous solution method [76] for reinitializing the states following a switching operation. However, when an electrical subsystem is interfaced with this software as a user-defined

component (as is done in this case for the DSE-based formulation), the program does not interpolate or reinitialize the internal states of the these components. Such a partial interpolation/reinitialization can lead to errors which are observed in this case.



(a) Dc Link Current (External Variable)



(b) Absolute Error in dc Link Current

Figure 2.10: Simulation Results for I_{dc} - Interfacing Example 2 (IEEE 39-bus)

Thus, the above two examples verify the accuracy of the proposed interfacing procedure presented earlier. They also demonstrate how the DSE-based approach can be used for interfacing arbitrary power networks with a commercial CC-based EMT simulation package.

As discussed in Section 2.4, partitioning the system and using the proposed interfacing approach permits the use of parallel computing techniques for speeding up the simulation. To demonstrate this feature, the two partitions (shown in Fig. 2.8) were simulated in parallel on a general-purpose Intel i7-8700 based PC with 6 cores running at 3.2 GHz and having Windows 10 OS. Table 2.5 shows a comparison of the CPU times for the parallel and serial simulation cases (here, serial simulation means the one that is not using parallel computing techniques). The times are for a 10 s simulation with a time-step $\Delta t = 20\ \mu s$.

Table 2.5: Comparison of Total CPU Times (IEEE 39 Bus System)

Serial Simulation T_S (sec)	Parallel Simulation T_P (sec)	Speedup (T_S/T_P)
161.25	93.15	1.73

As we can see in Table 2.5, partitioning the system into two roughly equally sized subsystems gave a speedup factor of 173%. Thus, this also verifies the fact that the proposed interfacing procedure can be used to speed up a simulation using parallel processing.

2.6 Conclusions

In this chapter, we investigated an alternative method to formulate state variable equations of a circuit for EMT simulations using Descriptor State-space Equations (DSE). A step-by-step procedure is presented for automatically formulating the DSEs using a circuit's netlist. The formulation is easier to implement as a computer algorithm compared to the classical state-space equations. Also, intermediate matrix manipulation steps are avoided in the DSE-based formulation thus making it suitable for large systems. Once formulated, the DSEs are discretized using trapezoidal integration method and used to carry out EMT simulations studies.

This approach is compared with the widely used Companion Circuits (CC) approach. One of the advantages of using the DSE-based formulation is that in addition to running EMT simulations, it is possible to analytically calculate the eigenvalues of the network directly. However, it has the disadvantage of having a marginally higher run-time than the CC-based approach for EMT simulation.

Finally, a procedure for interfacing a DSE-based formulation with a CC-based EMT simulator is also presented. There is no time-step delay involved while interfacing. This procedure enables interfacing of arbitrary power networks with a commercial CC-based EMT simulation package without the need for building it in that package. Moreover, this combined approach also allows easy parallel simulation as multiple DSE-based modules can be run on

separate processors and then interfaced with a CC-based EMT simulator.

2.7 Contributions

1. Investigated an alternative method for formulating state variable equations of a circuit for EMT simulations using Descriptor State-space Equations (DSE).

2. Presented a step-by-step procedure for automatically formulating the DSEs using a circuit's netlist.

3. Compared the computational performance of DSE-based EMT simulation with conventional Companion Circuits (CC)-based EMT simulation.

4. Developed a procedure for interfacing a DSE-based formulation with a CC-based EMT simulator that enables interfacing of arbitrary power networks with any commercial CC-based EMT simulation package and can also be used to speed up the simulation using parallel processing.

Chapter 3

Acceleration of DSE-based EMT Simulations using Matrix-level Parallelization Methods

This chapter describes two commonly used sparse matrix-based parallelization methods, and then adapts and utilizes these in order to accelerate DSE-based EMT simulations, and compares their computational performance. In the first method, a sparse matrix is transformed into a Block Diagonal (BD) form and can be used when the simulated network contains distributed parameter transmission lines and/or cables. Whereas, the second method transforms a sparse matrix into a Bordered Block Diagonal (BBD) form and is more general in its application.

3.1 Introduction

In Chapter 2, we investigated an alternative method for formulating the equations of a network for Electromagnetic Transient (EMT) simulations. This method uses Descriptor State-space Equations (DSE) for modelling the transient behavior of a circuit. In this approach, the equations of a circuit are firstly formulated in the continuous time domain using

Modified Nodal Analysis (MNA) in the form given in (3.1).

$$\mathbf{E}\dot{\underline{x}} = -\mathbf{A}\underline{x} + \mathbf{B}\underline{u} \tag{3.1}$$

Subsequently, (3.1) is discretized using the trapezoidal integration method with a time-step Δt to give (3.2) which is then solved in every time-step.

$$\left(\mathbf{E} + \frac{\mathbf{A}\Delta t}{2}\right)\underline{x}(t) = \left(\mathbf{E} - \frac{\mathbf{A}\Delta t}{2}\right)\underline{x}(t-\Delta t) + \frac{\mathbf{B}\Delta t}{2}\left(\underline{u}(t) + \underline{u}(t-\Delta t)\right) \tag{3.2}$$

For large-size networks (i.e., ones containing a large number of electrical nodes as well as a large number of inductors and independent voltage sources), the most computationally intensive step in DSE-based EMT simulations is the solution of (3.2) for computing $\underline{x}(t)$ in every time-step. One of the ways of reducing the CPU runtime of DSE-based simulations for such systems is to exploit the parallel processing capabilities offered by modern multi-core CPU systems. We can see that (3.2) is of the form:

$$\mathbf{M}\underline{x} = \underline{b} \tag{3.3}$$

where,

$$\mathbf{M} = \left(\mathbf{E} + \frac{\mathbf{A}\Delta t}{2}\right) \tag{3.4}$$

$$\underline{b} = \left(\mathbf{E} - \frac{\mathbf{A}\Delta t}{2}\right)\underline{x}(t-\Delta t) + \left(\frac{\mathbf{B}\Delta t}{2}\right)\left(\underline{u}(t) + \underline{u}(t-\Delta t)\right) \tag{3.5}$$

$\underline{x}$: Vector of unknowns (descriptor state variables at time t)

Thus, for speeding up the computation of $\underline{x}$ in every time-step using parallel processing, the following two tasks need to be accomplished:

(i) Speed up the computation of the right-hand side (RHS) vector $\underline{b}$.

(ii) Speed-up the solution of $\mathbf{M}\underline{x} = \underline{b}$.

From (3.5), it is clear that the computation of $\underline{b}$ only involves basic operations like matrix-

vector multiplication and vector addition. Hence, its computation can be sped up by using standard data partitioning strategies that are used for achieving parallelism in many parallel programming applications [36]. This will be discussed in detail in Section 3.3.

In contrast, parallelizing the solution of $\mathbf{M}\underline{x} = \underline{b}$ in order to compute $\underline{x}$ is comparatively more challenging. This is because this problem is usually solved by firstly factorizing the matrix $\mathbf{M}$ into a product of a lower and upper triangular matrix, and then using forward-backward substitution. There is no straightforward way of parallelizing the forward-backward substitution strategy.

However, for large-size networks, the matrix $\mathbf{M}$ is typically very sparse. This property of $\mathbf{M}$ can be leveraged to parallelize the solution of $\mathbf{M}\underline{x} = \underline{b}$ using sparse matrix-based parallelization methods. These methods essentially convert $\mathbf{M}$ to a form that is amenable to parallel processing. Several matrix-level parallelization methods have been proposed in the literature for accelerating CC-based EMT simulations [34, 46–48]. In this chapter, we will adapt and use two of the competing popular methods for speeding up DSE-based EMT simulations. The first method transforms $\mathbf{M}$ into a block diagonal form while the second one transforms $\mathbf{M}$ into a bordered block diagonal form. These two methods will be discussed in detail in Section 3.4.

The main aim of the work reported in this chapter is:

(a) To show that DSE-based EMT simulations can be accelerated using matrix-level parallelization methods.

(b) To compare the computational performance of the two matrix-level parallelization methods that will be used for accelerating DSE-based EMT simulations.

The work reported in this chapter uses a shared memory computing system (i.e., a single CPU with multiple cores, with the same memory space being accessible to each core) in order to facilitate the parallelization of DSE-based EMT simulations. OpenMP [77] (an Application Programming Interface (API) for shared memory parallel computing) is used with the Single Program Multiple Data (SPMD) paradigm [36] for achieving parallelism. Because of the shared memory architecture of the computing system that is used in this

work, explicit instructions for exchanging data between different processor cores are not required [34].

3.2 Overview of OpenMP

This section gives a very brief overview of OpenMP - the API used for parallel programming in this work. OpenMP is an API for shared memory parallel computing in C/C++ and Fortran [78]. It uses a combination of compiler directives, runtime library functions, and environment variables for implementing different parallel processing constructs like *work-sharing* constructs, *synchronization* constructs, etc. [36]. OpenMP uses a *fork-join* model (illustrated in Fig. 3.1) for parallelizing sections of a program using multi-threading.

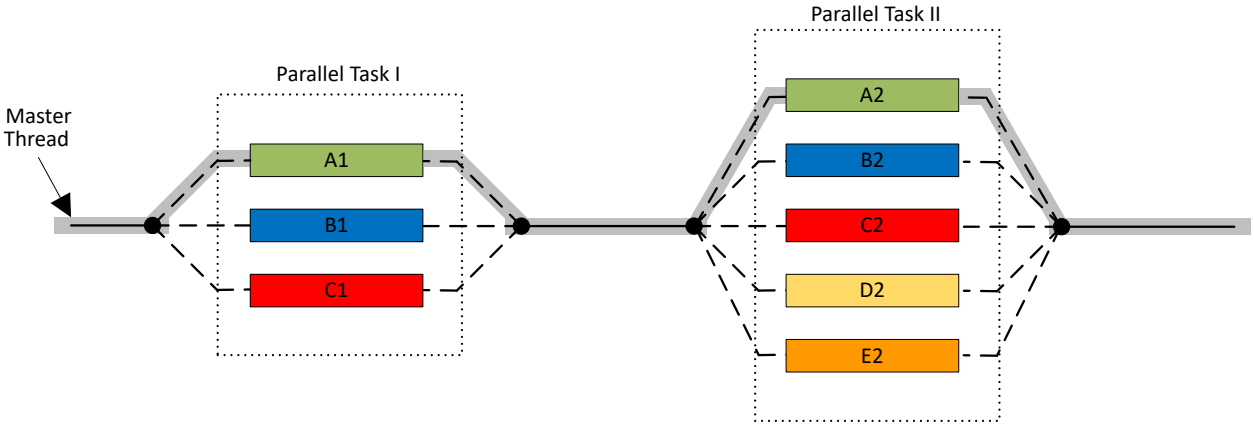


Figure 3.1: Fork-join model of OpenMP

In this approach, a *master* thread executes a code serially till it encounters a portion of the code to be run in parallel. At this point, the *master* thread *forks* multiple threads, and the task is divided among this team of threads (the *master* thread has rank = 0 in this team, shown by the grey line in Fig. 3.1). Different threads are allocated to different processors by the run-time environment of OpenMP so that they can run concurrently. Once the execution of the parallel portion is completed by the team of threads, they *join* back into the *master* thread after which the *master* thread continues the execution serially until the end of the program or until the next parallel section is encountered. For more details about OpenMP,

the reader is referred to [77, 78].

In this work, OpenMP is used in C++. The number of threads to be *forked* for every parallel section is specified at the start of the program itself by calling the run-time library function *omp_set_num_threads()*. Portions of the code to be run in parallel are included inside blocks enclosed by the *#pragma omp parallel* directive. Some of the other compiler directives of OpenMP that have been used in this work are:

(a) *#pragma omp for* - To parallelize the execution of *for* loops.

(b) *#pragma omp critical* - To avoid race conditions between different threads when they read from or write into the same variable.

3.3 Accelerating the Computation of RHS Vector $\underline{b}$

As discussed in Section 3.1, for accelerating the solution of (3.2) in DSE-based EMT simulations, the first step is to speed up the computation of the RHS vector $\underline{b}$. From (3.5), we can see that $\underline{b}$ can be expressed as:

$$\underline{b} = \underline{b}_1 + \underline{b}_2 \tag{3.6}$$

where,

$$\underline{b}_1 = \left(\mathbf{E} - \frac{\mathbf{A}\Delta t}{2}\right)\underline{x}(t - \Delta t) \tag{3.7}$$

$$\underline{b}_2 = \left(\frac{\mathbf{B}\Delta t}{2}\right)(\underline{u}(t) + \underline{u}(t - \Delta t)) \tag{3.8}$$

From (3.7) and (3.8), it is clear that both $\underline{b}_1$ and $\underline{b}_2$ are results of matrix-vector products, and only depend on history values ($\underline{x}(t - \Delta t)$ and $\underline{u}(t - \Delta t)$) from the previous time-step as well as the known source vector $\underline{u}(t)$ from the current time-step. Thus the computation of $\underline{b}$ can be accelerated by using the approach discussed next.

Parallelizing Sparse Matrix-Vector Product

One of the effective strategies for parallelizing a matrix-vector product is to use the *dot-product* approach [79] illustrated visually in Fig. 3.2. In this approach, for computing $\underline{y} = \mathbf{N}\underline{x}$, the result vector $\underline{y}$ is partitioned row-wise and each thread then computes its own portion of the result vector in parallel. This essentially means that the input matrix $\mathbf{N}$ should be partitioned row-wise among the team of threads while the input vector $\underline{x}$ should be shared across all threads.

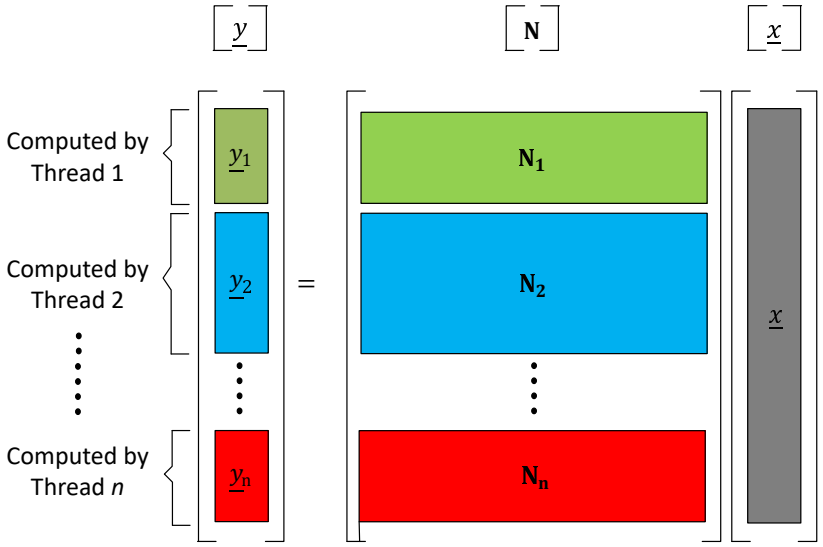


Figure 3.2: Dot-product approach for parallelizing matrix-vector product

Now while accelerating the computation of the RHS vector $\underline{b}$ in DSE-based EMT simulations, notice that the matrices $\left(\mathbf{E} - \frac{\mathbf{A}\Delta t}{2}\right)$ and $\mathbf{P} = \left(\frac{\mathbf{B}\Delta t}{2}\right)$ in (3.7) and (3.8) respectively, can both be typically very sparse for large-size systems. That implies that each row of these matrices may contain widely varying number of non-zero elements. If we use a naive approach of equally partitioning the number of rows of the result vectors ($\underline{b}_1$ and $\underline{b}_2$) among the team of threads, there is no guarantee that computational load balancing would be achieved. To overcome this problem, the rows of the result vector are partitioned in such a way that each individual chunk of the corresponding input matrix allotted to each thread has roughly an equal number of non-zero elements in them [79].

3.4 Accelerating the Solution of $\mathbf{M}\underline{x} = \underline{b}$

In this section, we will discuss two sparse matrix-based parallelization methods for accelerating the solution of $\mathbf{M}\underline{x} = \underline{b}$ in every time-step. In each of these methods, $\mathbf{M}$ is firstly converted to a form that is amenable to parallel processing. Following this, the corresponding (transformed) matrix is factorized into a product of a block-wise lower triangular matrix ($\mathbf{L}$) and a block-wise upper triangular matrix ($\mathbf{U}$). These are then used for solving $\mathbf{M}\underline{x} = \underline{b}$ using forward-backward substitution.

In this work, the *CSparse* library in C++ is used for all sparse matrix related operations including LU factorization [80]. Also, the Approximate Minimum Degree (AMD) algorithm [81] is used for computing a column permutation matrix prior to factorizing a sparse matrix so that the fill-ins created in its LU factors are reduced.

3.4.1 Method 1: M transformed to a Block Diagonal (BD) form

This section discusses the first method for accelerating the solution of $\mathbf{M}\underline{x} = \underline{b}$ in every time-step. In this method, the matrix $\mathbf{M}$ is transformed into a Block Diagonal (BD) form. This method can only be used when the network to be simulated contains distributed parameter transmission lines and/or cables[1].

BD Matrix and its LU Factorization

If the network to be simulated contains distributed parameter line/cable models, then by using an appropriate row permutation matrix $\mathbf{P}_{bd}$ and column permutation matrix $\mathbf{Q}_{bd}$, the matrix $\mathbf{M}$ can be transformed into a block diagonal matrix as shown in (3.9). This is because when the travel time of a line or a cable is greater than one time-step, it computationally decouples the networks at its two ends in a simulation, as the variations at one of its ends do not affect the other end in the same time-step [37].

[1]The procedure for including distributed parameter elements such as cables or transmission lines in a DSE-based formulation is given in Appendix B

In (3.9), N is the number of blocks and is equal to the number of sub-networks created by the decoupling enabled by the distributed parameter line/cable models [37].

$$\begin{aligned}\mathbf{M}_{bd} &= \mathbf{P}_{bd}\mathbf{M}\mathbf{Q}_{bd} \\ &= \begin{bmatrix} \mathbf{M}_{11} & & & \\ & \mathbf{M}_{22} & & \\ & & \ddots & \\ & & & \mathbf{M}_{NN} \end{bmatrix}\end{aligned} \tag{3.9}$$

Using (3.9), the $\mathbf{M}\underline{x} = \underline{b}$ problem in (3.2) transforms to (3.10).

$$\mathbf{M}_{bd}\underline{y} = \underline{c} \implies \begin{bmatrix} \mathbf{M}_{11} & & & \\ & \mathbf{M}_{22} & & \\ & & \ddots & \\ & & & \mathbf{M}_{NN} \end{bmatrix} \begin{bmatrix} \underline{y}_1 \\ \underline{y}_2 \\ \vdots \\ \underline{y}_N \end{bmatrix} = \begin{bmatrix} \underline{c}_1 \\ \underline{c}_2 \\ \vdots \\ \underline{c}_N \end{bmatrix} \tag{3.10}$$

Here,

$$\underline{c} = \mathbf{P}_{bd}\underline{b} \tag{3.11}$$

$$\underline{y} = \mathbf{Q}_{bd}^T\underline{x} \tag{3.12}$$

For solving (3.10), a block LU factorization of $\mathbf{M}_{bd}$ is performed as given in (3.13) with $\mathbf{L}$ and $\mathbf{U}$ as in (3.14).

$$\mathbf{M}_{bd} = \mathbf{L}\mathbf{U} \tag{3.13}$$

where,

$$\mathbf{L} = \begin{bmatrix} \mathbf{P}_{11}^T\mathbf{L}_{11} & & & \\ & \mathbf{P}_{22}^T\mathbf{L}_{22} & & \\ & & \ddots & \\ & & & \mathbf{P}_{NN}^T\mathbf{L}_{NN} \end{bmatrix};\ \mathbf{U} = \begin{bmatrix} \mathbf{U}_{11}\mathbf{Q}_{11}^T & & & \\ & \mathbf{U}_{22}\mathbf{Q}_{22}^T & & \\ & & \ddots & \\ & & & \mathbf{U}_{NN}\mathbf{Q}_{NN}^T \end{bmatrix} \tag{3.14}$$

In (3.14), every:

P: Permutation matrix corresponding to partial pivoting.

L: A lower triangular matrix.

U: An upper triangular matrix.

Q: Permutation matrix for reducing fill-ins in **L** and **U**.

As is clear from (3.10), every block equation is completely independent of the other. Hence, each of these sub-matrices in **L** and **U** can be obtained by using a standard LU factorization as shown in (3.15).

$$\mathbf{M}_{kk} = \mathbf{P}_{kk}^{T} \left(\mathbf{L}_{kk}\mathbf{U}_{kk}\right) \mathbf{Q}_{kk}^{T}, \quad k = 1, 2, \cdots N \tag{3.15}$$

Solving $\mathbf{M}_{bd}\underline{y} = \underline{c}$ using LU Factors of a BD Matrix

To compute the unknown vector $\underline{y}$, the following two-step procedure is used:

1. Solve (3.16) using forward substitution in order to compute the intermediate unknown variables $\underline{z}$.

$$\mathbf{L}\underline{z} = \underline{c} \tag{3.16}$$

   where, $\underline{z} = \begin{bmatrix} \underline{z}_1 & \underline{z}_2 & \cdots & \underline{z}_N \end{bmatrix}^T$

   From (3.10), it is clear that every block equation is completely independent of the other. Therefore, (3.16) is solved for individual $\underline{z}_k$'s using (3.17).

$$\mathbf{L}_{kk}\underline{z}_k = \mathbf{P}_{kk}\underline{c}_k, \quad k = 1, 2, \cdots N \tag{3.17}$$

2. Using the $\underline{z}$ computed using (3.16), compute $\underline{y}$ by solving (3.18) using backward substitution.

$$\mathbf{U}\underline{y} = \underline{z} \tag{3.18}$$

   Again, since every block equation is completely independent of the other, (3.18) is solved for individual $\underline{y}_k$'s using (3.19).

$$\mathbf{U}_{kk}\underline{w}_k = \underline{z}_k \quad \text{and} \quad \underline{y}_k = \mathbf{Q}_{kk}\underline{w}_k \quad k = 1, 2, \cdots N \tag{3.19}$$

Parallelization of DSE-based EMT Simulations using a BD Matrix

Fig. 3.3 presents a flowchart to illustrate how parallelization of DSE-based EMT simulations is possible using a BD matrix. The blocks labelled 'A', 'B' and 'C' is where parallelism can be implemented.

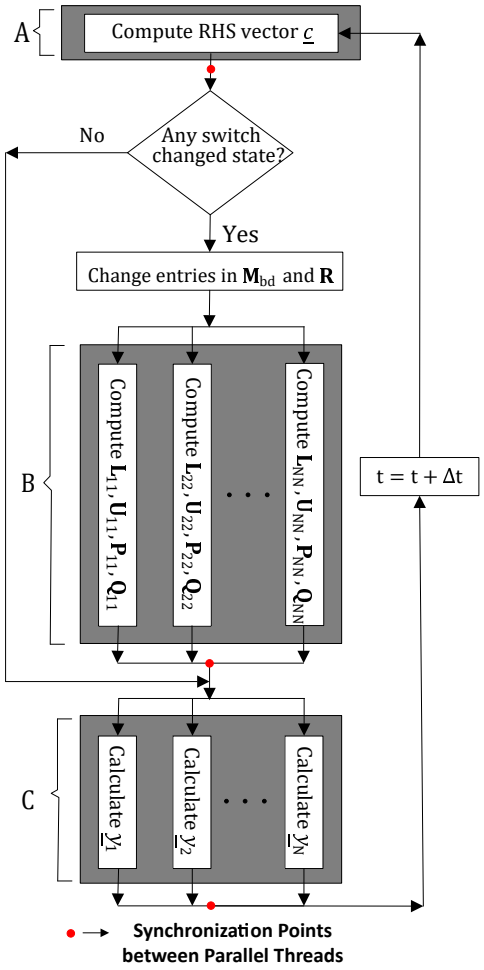


Figure 3.3: Parallelization of DSE-based EMT Simulations using a Block Diagonal Matrix

1. Block A: The computation of the RHS vector $\underline{c}$ can be parallelized using the approach discussed in Section 3.3.

2. Block B: When any switching element in the circuit changes its state, firstly the corresponding entries in $\mathbf{M}_{bd}$ and $\mathbf{R} = \left(\mathbf{E} - \frac{\mathbf{A}\Delta t}{2}\right)$ must be changed, and then $\mathbf{M}_{bd}$ must

be refactorized. By inspecting (3.15), it is clear that each $\mathbf{M}_{kk}$ can be refactorized independently in parallel.

3. Block C: From (3.16) - (3.19), it is clear that each $\underline{y}_k$ can be calculated independently in parallel.

It is worthwhile to note that usually for systems containing a large number of distributed parameters lines/cables, the number of blocks in $\mathbf{M}_{bd}$ (i.e., N) is much larger than the number of parallel threads (n_p) that are available. Moreover, the size of different blocks may be widely different. If a *static* scheduling strategy[2] is used for completing the tasks in Block 'C', then there is no guarantee that optimum load balancing would be achieved across processors. Hence in this work, a *work-pool* approach [36] is used for *dynamically* dividing the tasks in Block 'C'. This is done in the following way:

a) Firstly, a *work-pool* of tasks (in this case, calculating $\underline{y}_k$) is created. Each parallel thread is initially assigned one task for it to complete.

b) As and when a parallel thread finishes its assigned task, it picks up the next task in the *work-pool* until all tasks are exhausted.

Transforming M to a BD Form

For automatically transforming the matrix $\mathbf{M}$ to a block diagonal matrix $\mathbf{M}_{bd}$, the Block Triangular Factorization (BTF) algorithm is used in this work [45]. This algorithm first constructs an equivalent graph from the non-zero pattern of a sparse matrix. Then, it finds strongly connected components in this graph in order to determine the permutation matrix that will be used for transforming the input matrix to a block diagonal form [45].

An implementation of the BTF algorithm is available in the public domain at [82]. Functionally, the BTF algorithm library from [82] works as shown in Fig. 3.4.

[2]Static scheduling is where each parallel thread is assigned a roughly equal number of tasks at compile-time.

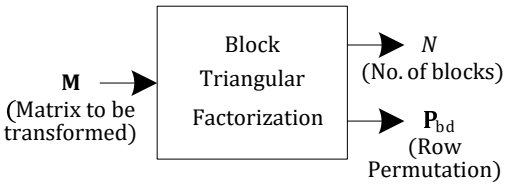


Figure 3.4: Converting to a Block Diagonal Matrix using the BTF Algorithm

It takes the sparse matrix to be transformed as its input and outputs the number of blocks N as well as the row permutation matrix $\mathbf{P}_{bd}$ that is required for transforming to a block diagonal matrix (see (3.9)). The column permutation matrix in (3.9) is $\mathbf{Q}_{bd} = \mathbf{P}_{bd}^T$.

3.4.2 Method 2: M transformed to a Bordered Block Diagonal (BBD) form

This section discusses the second method used for accelerating the solution of $\mathbf{M}\underline{x} = \underline{b}$ in every time-step. It transforms $\mathbf{M}$ into a Bordered Block Diagonal (BBD) form. This method is applicable in a more general case for a highly sparse matrix, i.e., it does not require the presence of any special elements like distributed parameter lines/cables in the simulated system; it uses the sparsity structure of the matrix for its transformation.

BBD Matrix and its LU Factorization

In general, any highly sparse matrix $\mathbf{M}$ can be converted to a Bordered Block Diagonal form by using an appropriate row permutation matrix $\mathbf{P}_{bbd}$ and column permutation matrix $\mathbf{Q}_{bbd}$, as shown in (3.20) [34].

$$\begin{aligned}\mathbf{M}_{bbd} &= \mathbf{P}_{bbd}\mathbf{M}\mathbf{Q}_{bbd} \\ &= \begin{bmatrix} \mathbf{M}_{11} & & & & \mathbf{M}_{1b} \\ & \mathbf{M}_{22} & & & \mathbf{M}_{2b} \\ & & \ddots & & \vdots \\ & & & \mathbf{M}_{nn} & \mathbf{M}_{nb} \\ \mathbf{M}_{b1} & \mathbf{M}_{b2} & \cdots & \mathbf{M}_{bn} & \mathbf{M}_{bb} \end{bmatrix}\end{aligned} \tag{3.20}$$

In (3.20), n is the number of blocks that are desired (i.e., its an input from the user) whereas $b = n + 1$. Using (3.20), the $\mathbf{M}\underline{x} = \underline{b}$ problem in (3.2) transforms to (3.21).

$$\mathbf{M}_{bbd}\underline{y} = \underline{c} \implies \begin{bmatrix} \mathbf{M}_{11} & & & & \mathbf{M}_{1b} \\ & \mathbf{M}_{22} & & & \mathbf{M}_{2b} \\ & & \ddots & & \vdots \\ & & & \mathbf{M}_{nn} & \mathbf{M}_{nb} \\ \mathbf{M}_{b1} & \mathbf{M}_{b2} & \cdots & \mathbf{M}_{bn} & \mathbf{M}_{bb} \end{bmatrix} \begin{bmatrix} \underline{y}_1 \\ \underline{y}_2 \\ \vdots \\ \underline{y}_n \\ \underline{y}_b \end{bmatrix} = \begin{bmatrix} \underline{c}_1 \\ \underline{c}_2 \\ \vdots \\ \underline{c}_n \\ \underline{c}_b \end{bmatrix} \tag{3.21}$$

Here,

$$\underline{c} = \mathbf{P}_{bbd}\underline{b} \tag{3.22}$$

$$\underline{y} = \mathbf{Q}_{bbd}^T\underline{x} \tag{3.23}$$

In (3.21), $\underline{y}_1$, $\underline{y}_2 \cdots \underline{y}_n$ represent the internal variables while $\underline{y}_b$ are the boundary variables which provide the coupling between the internal variables. For solving (3.21), a block LU factorization of $\mathbf{M}_{bbd}$ is performed as given in (3.24).

$$\mathbf{M}_{bbd} = \mathbf{L}\mathbf{U} \tag{3.24}$$

Because of the special structure of $\mathbf{M}_{bbd}$, the $\mathbf{L}$ and $\mathbf{U}$ matrices are as shown in (3.25) and (3.26) respectively [34].

$$\mathbf{L} = \begin{bmatrix} \mathbf{P}_{11}^T\mathbf{L}_{11} & & & & \\ & \mathbf{P}_{22}^T\mathbf{L}_{22} & & & \\ & & \ddots & & \\ & & & \mathbf{P}_{nn}^T\mathbf{L}_{nn} & \\ \mathbf{L}_{b1} & \mathbf{L}_{b2} & \cdots & \mathbf{L}_{bn} & \mathbf{P}_{bb}^T\mathbf{L}_{bb} \end{bmatrix} \tag{3.25}$$

$$\mathbf{U} = \begin{bmatrix} \mathbf{U}_{11}\mathbf{Q}_{11}^T & & & & \mathbf{U}_{1b} \\ & \mathbf{U}_{22}\mathbf{Q}_{22}^T & & & \mathbf{U}_{2b} \\ & & \ddots & & \vdots \\ & & & \mathbf{U}_{nn}\mathbf{Q}_{nn}^T & \mathbf{U}_{nb} \\ & & & & \mathbf{U}_{bb}\mathbf{Q}_{bb}^T \end{bmatrix} \tag{3.26}$$

Each of the sub-matrices in $\mathbf{L}$ and $\mathbf{U}$ in (3.25) and (3.26) respectively, are obtained using the following procedure:

1. The internal sub-matrices $\mathbf{P}_{kk}$, $\mathbf{L}_{kk}$, $\mathbf{U}_{kk}$ and $\mathbf{Q}_{kk}$ are obtained by a standard LU factorization as shown in (3.27).

$$\mathbf{M}_{kk} = \mathbf{P}_{kk}^T \left(\mathbf{L}_{kk}\mathbf{U}_{kk}\right) \mathbf{Q}_{kk}^T, \quad k = 1, 2, \cdots n \tag{3.27}$$

2. Once these are known, the border sub-matrices $\mathbf{L}_{bk}$ and $\mathbf{U}_{kb}$ can be obtained using (3.28) and (3.29) respectively.

$$\mathbf{L}_{bk} = \mathbf{M}_{bk}\mathbf{Q}_{kk}\mathbf{U}_{kk}^{-1}, \quad k = 1, 2, \cdots n \tag{3.28}$$

$$\mathbf{U}_{kb} = \mathbf{L}_{kk}^{-1}\mathbf{P}_{kk}\mathbf{M}_{kb}, \quad k = 1, 2, \cdots n \tag{3.29}$$

3. Finally, the sub-matrices $\mathbf{P}_{bb}$, $\mathbf{L}_{bb}$, $\mathbf{U}_{bb}$ and $\mathbf{Q}_{bb}$ at the lower-most block diagonal position ($b = n + 1$) are obtained by another LU factorization as shown in (3.30).

$$\left(\mathbf{M}_{bb} - \sum_{k=1}^{n} \mathbf{L}_{bk}\mathbf{U}_{kb}\right) = \mathbf{P}_{bb}^T \left(\mathbf{L}_{bb}\mathbf{U}_{bb}\right) \mathbf{Q}_{bb}^T, \quad k = 1, 2, \cdots n \tag{3.30}$$

In (3.27) and (3.30), every:

P: Permutation matrix corresponding to partial pivoting.

L: A lower triangular matrix.

U: An upper triangular matrix.

Q: Permutation matrix for reducing fill-ins in **L** and **U**.

Solving $\mathbf{M}_{bbd}\underline{y} = \underline{c}$ using LU Factors of a BBD Matrix

To solve (3.21) for computing the unknown vector $\underline{y}$, a forward-backward substitution procedure is used using the $\mathbf{L}$ and $\mathbf{U}$ matrices given in (3.25) and (3.26) respectively. This is done using the following two-step procedure is used:

1. Solve (3.31) using forward substitution in order to compute the intermediate unknown variables $\underline{z}_k$ for $k = 1, 2, \cdots N$.

$$\mathbf{L}\underline{z} = \underline{c} \tag{3.31}$$

   where, $\underline{z} = \begin{bmatrix}\underline{z}_1 & \underline{z}_2 & \cdots & \underline{z}_n & \underline{z}_b\end{bmatrix}^T$.

   Eq. (3.31) is solved by first solving for the intermediate internal variables $\underline{z}_k$'s in (3.32) and then using these $\underline{z}_k$'s in (3.33) to solve for the intermediate border variable vector $\underline{z}_b$.

$$\mathbf{L}_{kk}\underline{z}_k = \mathbf{P}_{kk}\underline{c}_k, \quad k = 1, 2, \cdots n \tag{3.32}$$

$$\mathbf{L}_{bb}\underline{z}_b = \mathbf{P}_{bb}\left(\underline{c}_b - \sum_{k=1}^{n}\mathbf{L}_{bk}\underline{z}_k\right) \tag{3.33}$$

2. Using $\underline{z}$ computed using (3.31), compute $\underline{y}$ by solving (3.34) using backward substitution.

$$\mathbf{U}\underline{y} = \underline{z} \tag{3.34}$$

   Eq. (3.34) is solved by first solving for the border variables $\underline{y}_b$ using (3.35) and then using it to solve for the internal variables $\underline{y}_k$'s using (3.36).

$$\mathbf{U}_{bb}\underline{w}_b = \underline{z}_b \quad \text{and} \quad \underline{y}_b = \mathbf{Q}_{bb}\underline{w}_b \tag{3.35}$$

$$\mathbf{U}_{kk}\underline{w}_k = \left(\underline{z}_k - \mathbf{U}_{kb}\underline{y}_b\right) \quad \text{and} \quad \underline{y}_k = \mathbf{Q}_{kk}\underline{w}_k \quad \text{where } k = 1, 2, \cdots n \tag{3.36}$$

Parallelization of DSE-based EMT Simulations using a BBD Matrix

Fig. 3.5 gives a flowchart to illustrate how parallelization of DSE-based EMT simulations is possible using a BBD matrix. The blocks labelled 'A', 'B', 'C' and 'D' is where parallelism

can be implemented.

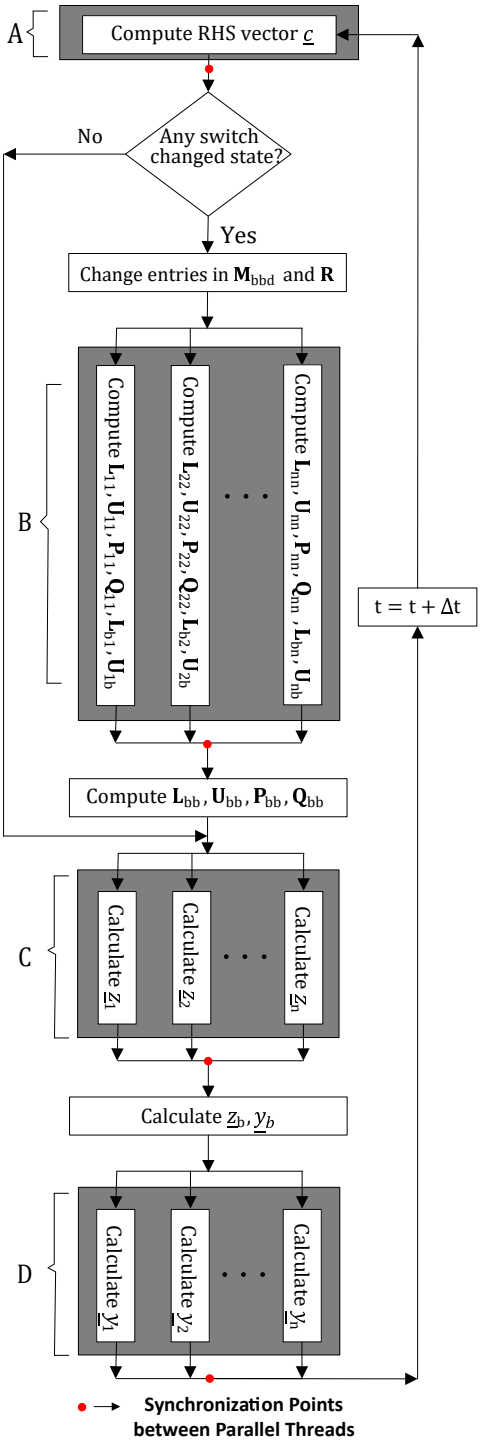


Figure 3.5: Parallelization of DSE-based EMT Simulations using a Bordered Block Diagonal Matrix

1. Block A: The computation of the RHS vector $\underline{c}$ can be parallelized using the approach discussed in Section 3.3.

2. Block B: When any switching element in the circuit changes its state, firstly the corresponding entries in $\mathbf{M}_{bbd}$ and $\mathbf{R} = \left(\mathbf{E} - \frac{\mathbf{A}\Delta t}{2}\right)$ must be changed, and then $\mathbf{M}_{bbd}$ must be refactorized. By inspecting (3.27), it is clear that each $\mathbf{L}_{kk}$, $\mathbf{U}_{kk}$, $\mathbf{P}_{kk}$ and $\mathbf{Q}_{kk}$ can be calculated independently in parallel. Similarly, from (3.28) and (3.29), it is clear that each $\mathbf{L}_{bk}$ and $\mathbf{U}_{kb}$ can also be computed in parallel. However, the computation of $\mathbf{L}_{bb}$, $\mathbf{U}_{bb}$, $\mathbf{P}_{bb}$ and $\mathbf{Q}_{bb}$ is a serial task which requires all the parallel threads to finish their prior computations (see (3.30)). Hence, the need for a synchronization point at the end of Block 'B'.

3. Block C: From (3.32), it is clear that each $\underline{z}_k$ (where $k = 1, 2, \cdots n$) can be calculated independently in parallel. However, computation of $\underline{z}_b$ and $\underline{y}_b$ is a serial task as is clear from (3.33) and (3.35). Hence, a synchronization point needs to be inserted at the end of Block 'C'.

4. Block D: Once all the $\underline{z}_k$'s (where $k = 1, 2, \cdots n$) as well as $\underline{z}_b$ and $\underline{y}_b$ are known, the computation of internal variables $\underline{y}_k$'s can be done in parallel (as is clear from (3.36)).

When converting $\mathbf{M}$ to $\mathbf{M}_{bbd}$, the number of internal blocks in the BBD form (i.e., n) is chosen to be equal to the number of parallel threads (n_p) that are available. Hence, for completing the tasks in Blocks 'C' and 'D', a *static* scheduling strategy [36] is used.

Transforming M to a BBD Form

In this work, the METIS graph partitioning library [44] is used for automatically converting a highly sparse matrix $\mathbf{M}$ to a bordered block diagonal matrix $\mathbf{M}_{bbd}$. The algorithms in METIS essentially use four steps - graph coarsening, initial partitioning, uncoarsening, and refinement - in order to produce roughly equally sized partitions of a graph with a minimum number of edges between these partitions [44].

The METIS library is available in the public domain at [83]. However, unlike the BTF library from [82], the METIS library does not directly accept sparse matrices as its input. One has to first construct an equivalent graph from the non-zero pattern of a sparse matrix in order for it to be given as input to the METIS library for partitioning. Moreover, to make sure that all the sub-matrices ($\mathbf{M}_{kk}$) on the main block diagonal of $\mathbf{M}_{bbd}$ are non-singular, it is essential that the original matrix $\mathbf{M}$ has a zero-free main diagonal[3]. Hence, the procedure illustrated by the flowchart in Fig. 3.6 is used for converting a general sparse matrix $\mathbf{M}$ to a bordered block diagonal matrix $\mathbf{M}_{bbd}$.

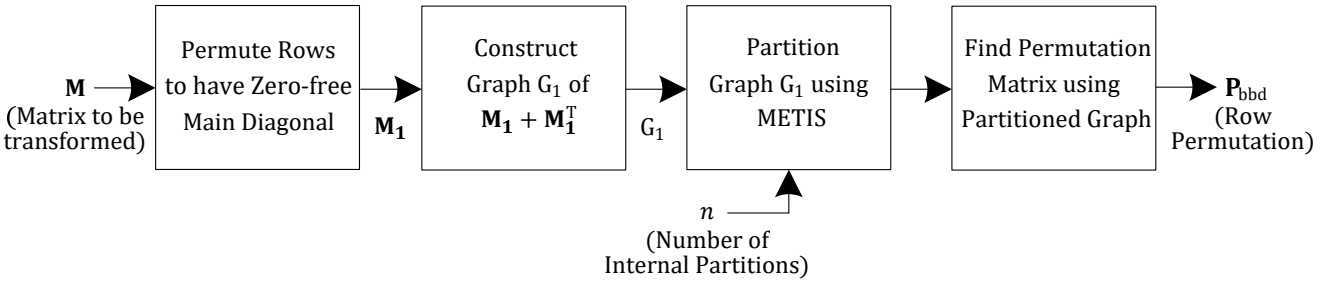


Figure 3.6: Converting to a Bordered Block Diagonal Matrix using METIS Graph Partitioning Library

An example is given in Appendix C to illustrate how this procedure works.

3.5 Simulation Test Cases

For comparing the computational performance of the two matrix-level parallelization methods discussed in Section 3.4, we consider two test cases in this section. The first is a 500-bus test system while the second is a 2000-bus test system. Both these test systems are synthetic grid cases based on the footprints of different regions in the USA [67]. The reason to use these "synthetic" test systems is to show the applicability of the method to large sized networks. Also, the data for these networks is not proprietary (available in the public domain at [68]) but the systems have been designed to show characteristics similar to real world networks.

[3]Since MNA is used in DSE-based formulation, if ideal voltage sources are present in the network, it can be easily shown that the matrix $\mathbf{M} = (\mathbf{E} + \mathbf{A}\Delta t/2)$ will end up with zeroes on its main diagonal.

The hardware and software configuration of the computing platform used for the computational performance comparison is given in Table 3.1.

Table 3.1: Hardware and Software Configuration of the Computing Platform

Processor	Intel Core i7-2600
# of Cores	4
Clock Speed	3.40 GHz
Memory	8 GB
Operating System	64-bit CentOS Linux 7

3.5.1 Test Case 1 - A 500 Bus System

The first test case is a 500-bus synthetic grid case based on the footprint of the network in the state of South Carolina, USA [67]. The data for this system has been taken from [68]. The system contains the following three-phase components: 466 transmission lines, 131 transformers, 206 loads (modelled as constant impedance loads), 15 capacitive shunts, and 56 voltage sources. A 10 s simulation is conducted with a time-step Δt = 50 μs. Two cases are considered for this example:

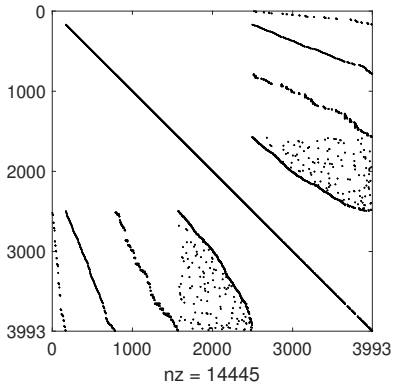
(a) Case 1: 159 out of the total 466 transmission lines are modelled using the Bergeron model [84]; the rest are modelled as coupled π-sections.

(b) Case 2: All 466 transmission lines are modelled as coupled π-sections.

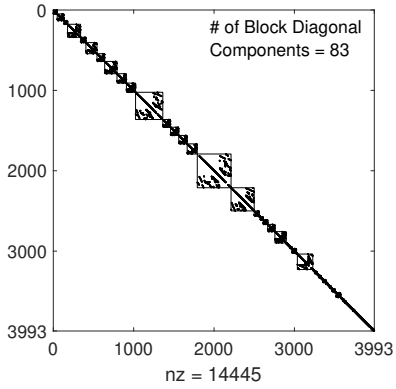
Case 1 - Containing both Distributed Parameter Lines and π-section Lines

Fig. 3.7a shows the non-zero pattern of the $\mathbf{M}$ matrix used in the serial simulation case. As distributed parameter lines are present in the simulated system, Method 1 can be used in this case. Hence, Fig. 3.7b shows the non-zero pattern of $\mathbf{M}_{bd}$ used in Method 1. There are a total of 83 block diagonal components in $\mathbf{M}_{bd}$ in this case. Additionally, as discussed earlier, Method 2 is more general in its application and hence can be readily used in this case. Hence, Figs. 3.7c-e show the non-zero pattern of $\mathbf{M}_{bbd}$ used in Method 2. Note that

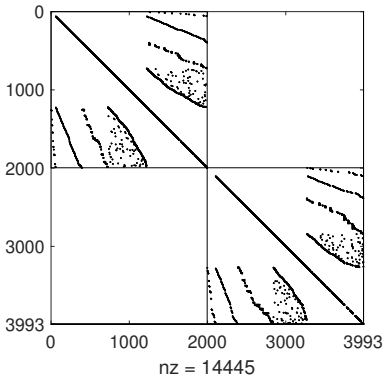
the size of the border blocks is zero in each of the $\mathbf{M}_{bbd}$'s due to the presence of distributed parameter lines (even though this information is not directly used while converting $\mathbf{M}$ to $\mathbf{M}_{bbd}$).



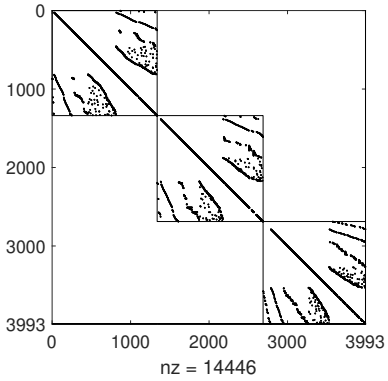
(a) M for Serial Simulation



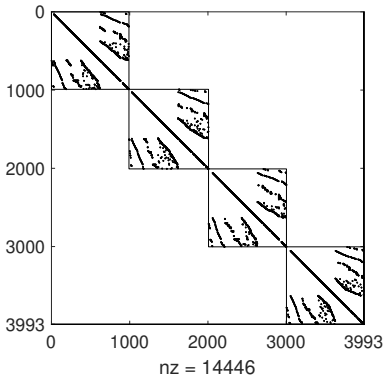
(b) $\mathbf{M}_{bd}$



(c) $\mathbf{M}_{bbd}$ ($n = 2$)



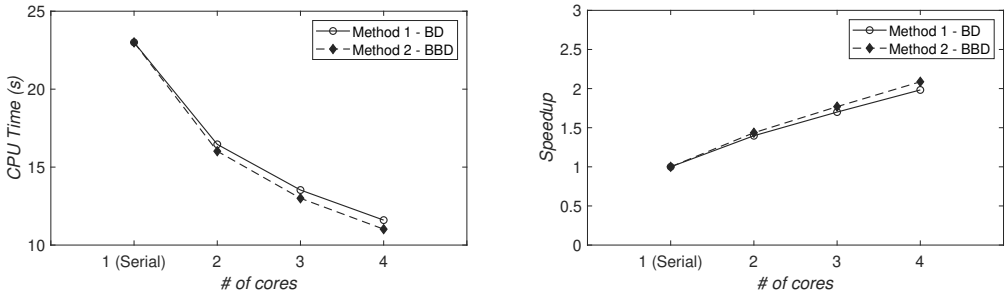
(d) $\mathbf{M}_{bbd}$ ($n = 3$)



(e) $\mathbf{M}_{bbd}$ ($n = 4$)

Figure 3.7: Sparsity patterns of matrices in Example 1 - Case 1

Fig. 3.8 presents the CPU runtime and the speedup factor as a function of the number of processor cores used for parallelization in both Method 1 and Method 2. As can be seen in Fig. 3.8, the speedup achieved using both methods is almost similar in this case.



(a) CPU Time **(b) Speedup**

Figure 3.8: Performance Plots for 500-bus Test System - Case 1

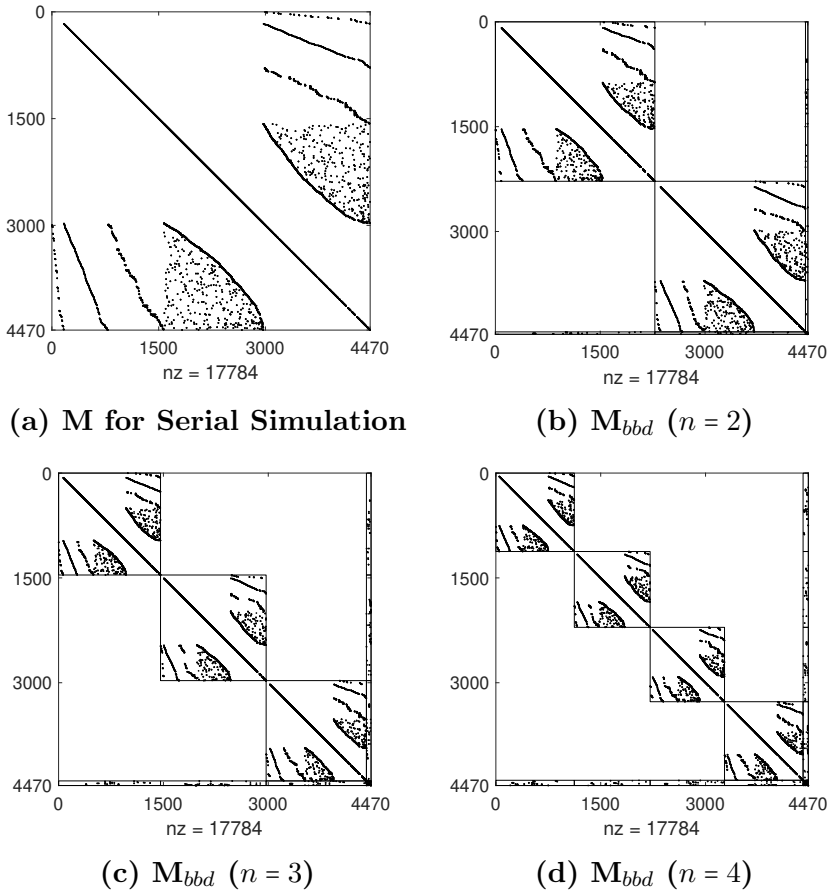
Case 2 - Containing π-section Lines Only

Since all the transmission lines in this case are modelled as coupled π-sections, there is no natural decoupling available to transform the matrix $\mathbf{M}$ to a block-diagonal (BD) form. Hence, Method 1 cannot be used in this case. However, Method 2 does not rely on any intrinsic decoupling provided by distributed parameter elements in the system. Hence, it can be readily used for parallelizing the simulation in this case.

Fig. 3.9a shows the non-zero pattern of the $\mathbf{M}$ matrix for the serial simulation case while Figs. 3.9b-d show the non-zero pattern of the BBD matrix $\mathbf{M}_{bbd}$. As is apparent from these figures, the size of the border blocks is *not* zero in this case as only π-section lines are present in this system. However, as seen in Figs. 3.9b-d, the size of the border blocks is significantly smaller compared to the internal blocks.

Fig. 3.10 shows the CPU runtime as well as the speedup as a function of the number of processor cores. Based on the performance plots in Fig. 3.10, we can conclude that Method 2 can be used for speeding up the simulation even when there are no elements that provide intrinsic decoupling in the simulated system. However, the speedup seems to be plateauing as the number of processors increase. We could not go beyond four threads because of the

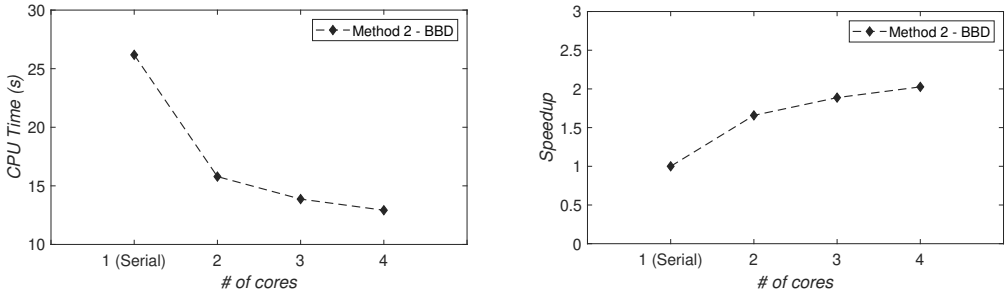
hardware constraints of the computing platform used in this work (see Table 3.1). As future work, we plan to test the scalability of this approach on a computing platform that has higher number of processor cores.



(a) M for Serial Simulation **(b) $\mathbf{M}_{bbd}$ ($n = 2$)**

(c) $\mathbf{M}_{bbd}$ ($n = 3$) **(d) $\mathbf{M}_{bbd}$ ($n = 4$)**

Figure 3.9: Sparsity patterns of matrices in Example 1 - Case 2



(a) CPU Time **(b) Speedup**

Figure 3.10: Performance Plots for 500-bus Test System - Case 2

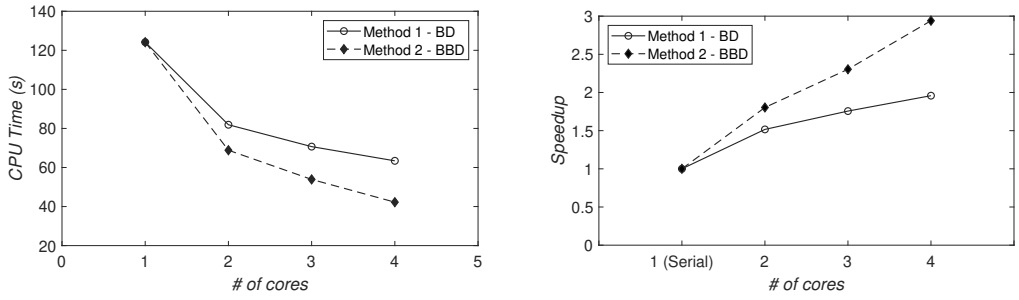
3.5.2 Test Case 2 - A 2000 Bus System

The second test case is a 2000-bus test system. It is a synthetic grid case based on the footprint of the network in Texas, USA [67]. The data for this system has been taken from [68]. The main objective of this test case is to compare the performance of the two sparse matrix-based parallelization methods in a relatively large system.

The test system contains the following three-phase components: 2481 transmission lines, 562 transformers, 1417 loads (modelled as constant impedance type), 36 capacitive shunts, and 282 voltage sources. 1406 out of the total 2481 transmission lines are modelled using the Bergeron model [84]; the rest are modelled as coupled π-sections. A 10 s simulation is conducted with a time-step $\Delta t = 50\ \mu s$.

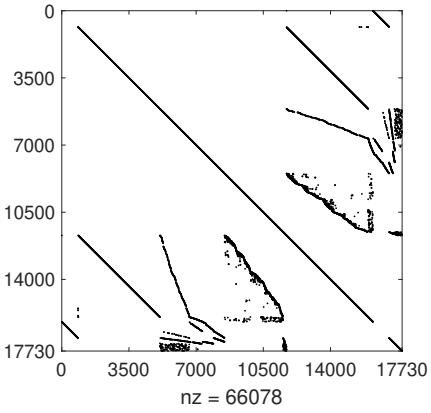
Fig. 3.12a shows the non-zero pattern of the $\mathbf{M}$ matrix used in the serial simulation case. Fig. 3.12b shows the non-zero pattern of $\mathbf{M}_{bd}$ used in Method 1 whereas Figs. 3.12c-e show the non-zero pattern of $\mathbf{M}_{bbd}$ used in Method 2.

Fig. 3.11 presents the performance plots for this case. In this case, Method 2 has far superior performance compared to Method 1. In Fig. 3.11b we can see that the speedup using Method 1 plateaus very quickly. On the other hand, Method 2 is able to provide an almost linear speedup with 2 cores and the speedup achieved with 4 cores is almost double that with Method 1.

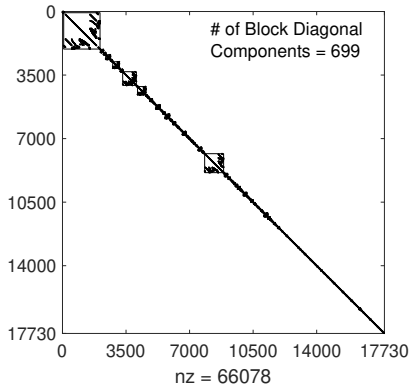


(a) CPU Time **(b) Speedup**

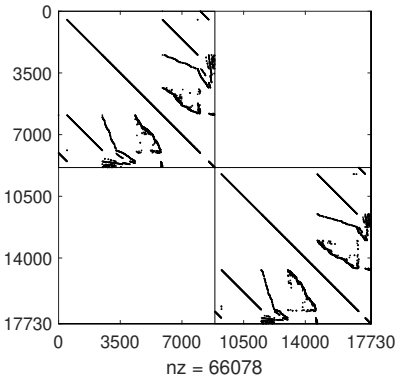
Figure 3.11: Performance Plots for 2000-bus Test System



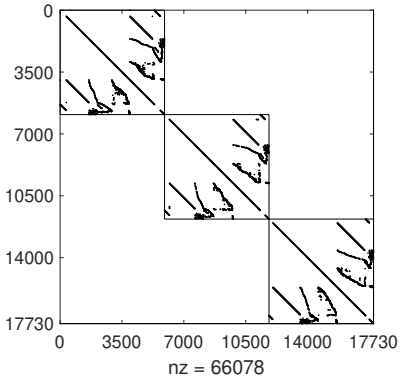
(a) M for Serial Simulation



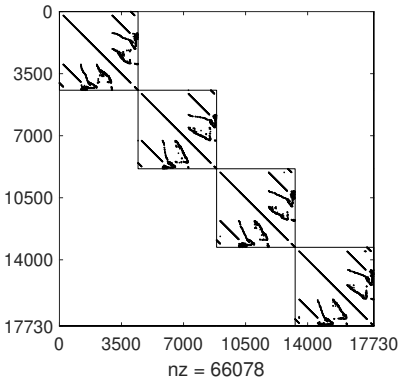
(b) $\mathbf{M}_{bd}$



(c) $\mathbf{M}_{bbd}$ ($n = 2$)



(d) $\mathbf{M}_{bbd}$ ($n = 3$)



(e) $\mathbf{M}_{bbd}$ ($n = 4$)

Figure 3.12: Sparsity patterns of matrices in Example 2

One of the probable reasons for the plateauing of the speedup with Method 1 is the overhead incurred while *dynamically* scheduling the forward-backward substitution tasks

while solving $\mathbf{M}\underline{x} = \underline{b}$.

3.6 Conclusions

In this chapter, two of the commonly used sparse matrix-based parallelization methods were adapted and utilized for accelerating DSE-based EMT simulations. Both methods relied on transforming the sparse matrix $\mathbf{M} = \left(\mathbf{E} + \frac{\mathbf{A}\Delta t}{2}\right)$ to a form which is amenable to parallel processing. In Method 1, $\mathbf{M}$ was automatically transformed to a block-diagonal form. This method can only be used when the simulated system contains distributed parameter lines and/or cables. On the other hand, Method 2 automatically transformed $\mathbf{M}$ to a bordered block diagonal form. This method is more general in its application as it only uses the sparsity pattern of the matrix for its transformation (i.e., it does not require the presence of any special elements like distributed parameter lines/cables in the simulated system).

For comparing the computational performance of these two methods, two test cases based on the footprints of real-world systems were used. The first case was a 500-bus test system while the second one was a 2000-bus test system. Comparative studies showed that Method 2 was able to provide higher simulation speedup.

3.7 Contributions

1. Adapted and utilized two commonly used matrix-level parallelization methods (viz., the first which transforms a sparse matrix to a Block Diagonal (BD) form [47], and the second which transforms it to a Bordered Block Diagonal (BBD) form [34]) for accelerating DSE-based EMT simulations.

2. Presented a detailed procedure for automatically transforming any highly sparse matrix to a BBD form with the resulting matrix always guaranteed to have non-singular diagonal blocks (refer to Section 3.4.2 and Appendix C for this).

3. Using realistic test cases, demonstrated that DSE-based EMT simulations can be ac-

celerated using matrix-level parallelization methods and compared the computational performance of the two matrix-level parallelization methods.

Chapter 4

Novel Universally Passive Delay-based Interface for Network Partitioning in EMT Simulations

This chapter presents a novel Universally Passive Network Partitioning (UPNP) interface for EMT simulations that uses existing inductors in the circuit to partition the network. The proposed delay-based interface is guaranteed to be passive which benefits the numerical stability of the simulation and also has superior performance compared to existing inductor-based partitioning approaches. It allows for simulation speed up when the solution of the partitioned network is computed on a parallel computing platform.

4.1 Introduction

In Chapter 2, an algorithm for interfacing a DSE-based formulation with a CC-based EMT simulator was presented which was then used for speeding up EMT simulations using parallel processing. A closer look at this interfacing procedure reveals that it is a type of diakoptics-based partitioning approach (discussed in Section 1.4.1). Parallelizing a simulation using a diakoptics-based partitioning approach can be computationally efficient only if the number of

interconnecting links between various subsystems is relatively small compared to the number of internal electrical nodes in each subsystem. If this is not the case for any system, then the performance improvements accrued through the use of parallel processing using diakoptics-based partitioning would not be significant (refer to the discussion in Section 1.4.1 to know why this happens).

An alternative to this is to use a delay-based partitioning approach. This approach relies on the finite time delay introduced by an element in the circuit for partitioning the system, with the resulting sub-networks only exchanging the interface voltages and currents from the previous time-steps at the end of each time-step. Owing to this, the admittance matrix of the network becomes block-diagonal [37] (as was shown in Section 1.4.1, Fig. 1.2) thereby enabling parallelism as each block can be solved on a separate computing node.

The delay-based partitioning approach is popular in many commercial EMT simulation packages available today [47, 71, 85, 86] because it is computationally simpler to implement than a diakoptics-based approach and has minimal serial computation steps. In most cases, natural delays in the system can be exploited to partition the system (such as the travel time across a transmission line [37]). However, a transmission line may not be always present at a convenient partitioning point that results in equitable sharing of the computational load. In such cases, some authors have proposed using an inductor to partition the system [3, 39, 40]. This technique is sufficiently general and has been used before [3, 39, 40] because of the abundance of inductive elements in power systems (e.g., transformer leakage, etc.). In one approach, an inductor is approximated as a transmission (stub) line which automatically introduces a delay in the system [39, 40]. In another approach, delay-based partitioning is achieved by discretizing the inductor's differential equation using an explicit integration method [3]. While the latter approach can give rise to numerically unstable simulations, the prior approach exhibits sharp poorly damped resonance peaks in the high frequency range which can hamper its accuracy.

To overcome these limitations, we develop a novel delay-based interface in this chapter that uses existing inductors in the circuit to partition the network. It is shown that the

accuracy and stability of the proposed interface is superior to that of existing delay-based approaches. The novelty of the proposed approach is that it is guaranteed to be passive, which benefits the numerical stability of a simulation. Moreover, it is shown, using per-unitized frequency response plots as well as time-domain simulations, that the proposed interface more closely matches the response of the original inductor compared to an equivalent stub-line interface.

4.2 Existing Methods for Delay-based Partitioning and their Limitations

4.2.1 Transmission line-based Partitioning

Distributed parameter transmission line-based partitioning is the most widely used delay-based partitioning technique and has been used from the earliest EMT programs [37]. As it uses the natural travel time of the line for partitioning, there is no loss of accuracy in this case. However, in the absence of a transmission line at a convenient partitioning point, previous researchers have proposed the approaches listed below.

4.2.2 Partitioning using Explicit Method-based Models

In this technique, the differential equation of an inductor or capacitor at the partitioning location is discretized using an explicit numerical integration method (e.g., Forward Euler). With such methods, the variable being integrated (i.e., voltage or current) in any time-step requires only the values of the variables from the previous time-step. This allows the system to be partitioned as shown in Fig. 4.1.

One such approach [3] uses an explicit central difference formula to discretize the differential equation of an inductor as in (4.1). Approximating $v_L(t - \Delta t/2)$ on the right hand side in (4.1) using a linear extrapolation that utilizes the previous two known values of the

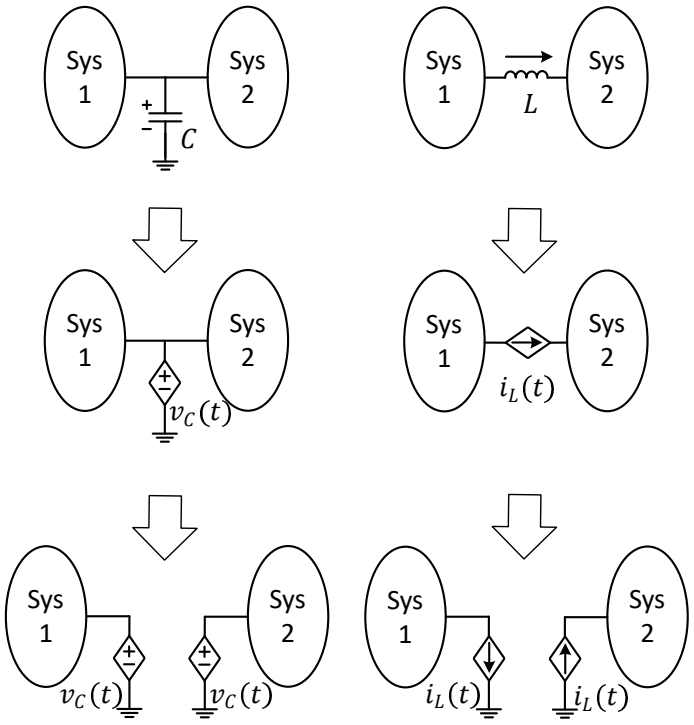


Figure 4.1: Partitioning using Explicit Method based Models

inductor voltage v_L [3] gives the relationship in (4.2).

$$i_L(t) = i_L(t-\Delta t) + \frac{\Delta t}{L} v_L\left(t - \frac{\Delta t}{2}\right) \tag{4.1}$$

$$\approx i_L(t-\Delta t) + \frac{\Delta t}{2L}\left(3v_L\left(t-\Delta t\right) - v_L\left(t-2\Delta t\right)\right) \tag{4.2}$$

A similar partitioning strategy can also be used with a shunt capacitor [3]. Although this technique gives a numerically stable result for the examples presented in [3], a stable simulation is not universally guaranteed as shown by the counter example given below:

Counter Example

Consider the circuit shown in Fig. 4.2. It is simulated with $\Delta t = 35\ \mu s$. The $0.1\ H$ inductor in Fig. 4.2 is modelled using the approach proposed in [3]. The switch is initially in the closed position with the circuit operating in steady state and then it is opened at $t = 0.05\ s$. The simulation result for the voltage V_2 is as shown in Fig. 4.3.

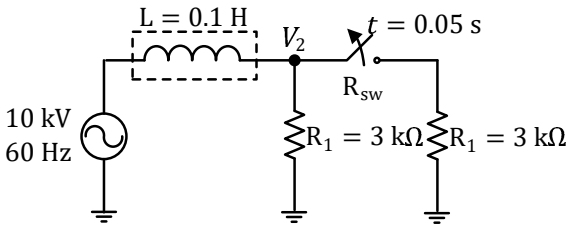


Figure 4.2: Circuit to demonstrate numerical instability with the partitioning approach proposed in [3]

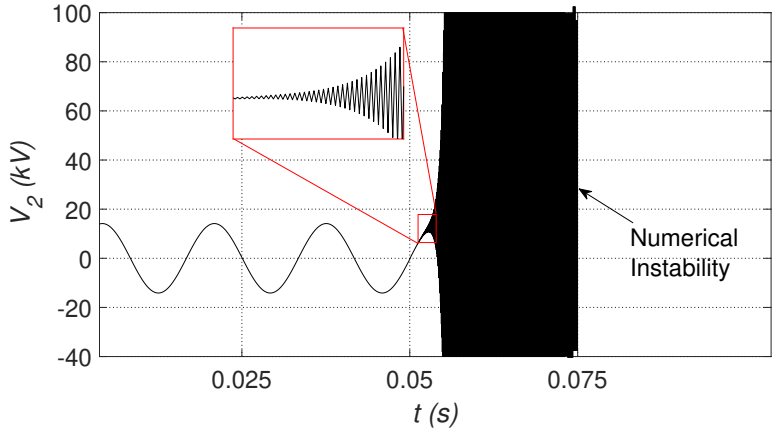


Figure 4.3: Numerical instability with partitioning approach proposed in [3]

We can see that the simulation is stable with the switch in the closed position. However, when the switch is opened, the response blows up indicating numerical instability. This instability can also be mathematically verified by computing the eigenvalues of the discretized system and checking if they lie inside or outside the unit circle for the case when the switch is open [65]. The general expression for the eigenvalues of the discretized equivalent circuit of Fig. 4.2 is given by:

$$\lambda_{1,2} = \frac{(2L - 3R\Delta t) \pm 2\sqrt{L^2 - RL\Delta t + \frac{9R^2\Delta t^2}{4}}}{4L}$$

Here, $R = R_1 \left(R_{sw} + R_1\right) / \left(R_{sw} + 2R_1\right)$.

When the switch is open, $R_{sw} \to \infty$ which implies $R = R_1$. With the values of R_1 and L as in Fig. 4.2, a plot of the eigenvalue magnitudes as a function of Δt is shown in Fig. 4.4.

From this figure, it is clear that the simulation using the approach proposed in [3] for the switch open case will always be unstable for $\Delta t > 33.4\ \mu s$. This is consistent with the result shown in Fig. 4.3 where $\Delta t = 35\ \mu s$ has been used.

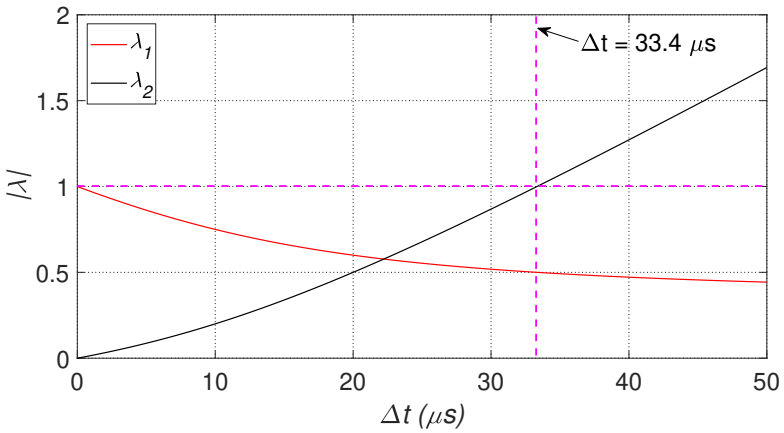


Figure 4.4: Eigenvalues as a function of Δt (Switch Open Case)

Thus, this counter example corroborates the fact that the approach proposed in [3] can indeed give rise to numerical instability in simulations.

4.2.3 Stub-line Interface

In this approach, an inductor at the partitioning location is represented using a fictitious "stub-line" with a one time-step (Δt) travel-time [39, 40]. Small fictitious capacitances to ground are introduced that enables the inductor to be represented by a Bergeron transmission line model as shown in Fig. 4.5 with characteristic impedance $Z_C = L/\Delta t$ and history current sources $i_k(t-\Delta t)$ and $i_m(t-\Delta t)$ as in (4.3). As these current sources depend only on history values of currents and voltages at either end, it allows for partitioning the system.

$$\begin{aligned} i_k(t-\Delta t) &= -i_2(t-\Delta t) - \frac{1}{Z_C} v_2(t-\Delta t) \\ i_m(t-\Delta t) &= -i_1(t-\Delta t) - \frac{1}{Z_C} v_1(t-\Delta t) \end{aligned} \tag{4.3}$$

The stub-line interface is inherently passive and so does not give rise to numerical instability in simulations. However, the introduction of the fictitious capacitances to ground can

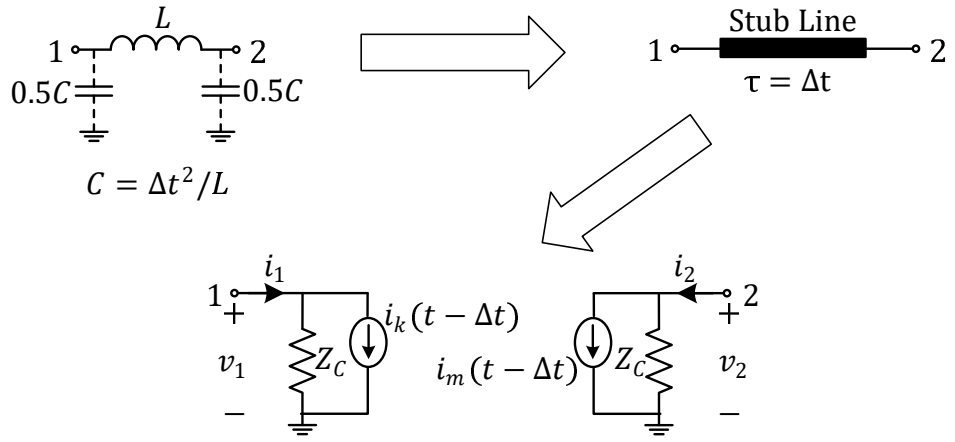


Figure 4.5: Stub-line Interface

introduce a resonant condition with the L and thus hamper its accuracy (as demonstrated in Section 4.4, 4.5 and 4.6).

To overcome the limitations of the above approaches, we propose a novel delay-based interface in this chapter that uses existing inductors in the circuit to partition the network. This interface is guaranteed passive which benefits the numerical stability of a simulation. Using per-unitized frequency response plots as well as time-domain simulations, it is also shown that the proposed interface more closely matches the response of the original inductor compared to an equivalent stub-line interface.

4.3 New Approach: Universally Passive Network Partitioning (UPNP) Interface

4.3.1 Development of the Proposed UPNP Interface

Consider the inductance L as in Fig. 4.6. First, L is split into two half-valued inductors ($L/2$) in series. Subsequently, similar to the stub-line interface described in Section 4.2.3, a 'fictitious' C branch is connected between the common node and ground (as shown in Fig. 4.6). If this capacitor C is discretized using the Forward Euler method (while retaining

the trapezoidal integration method for discretizing the inductors), then it transforms to just a history voltage source $V_h(t)$ as shown in Fig. 4.7a. This allows for partitioning the system as shown in Fig. 4.7b.

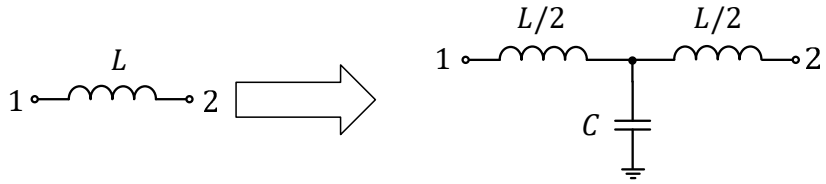


Figure 4.6: First step in the development of UPNP interface

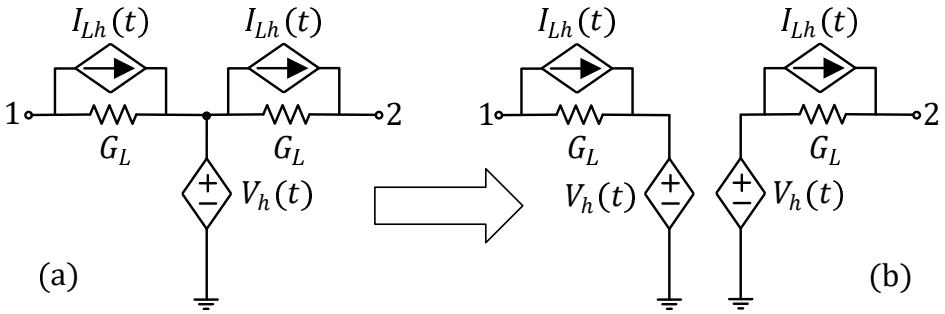


Figure 4.7: a) Discretized equivalent for the circuit in Fig. 4.6; b) partitioned circuit

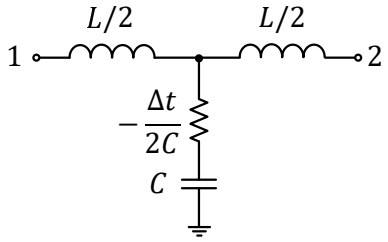


Figure 4.8: Continuous Time equivalent for circuit in Fig. 4.7 when all the elements are discretized using the trapezoidal method

However, it can be readily shown that the partitioned circuit in Fig. 4.7b is mathematically identical to the companion circuit that would result if the circuit in Fig. 4.8 were to be discretized using the trapezoidal integration method only (refer to Appendix D for the proof). And we can see that the circuit in Fig. 4.8 is not passive (due to the presence of the $-\Delta t/(2C)$ resistor, where Δt is the time-step used for the simulation). Hence, based on

the results in [65], we can conclude that the resulting simulation when using the partitioned circuit of Fig. 4.7b is *not* guaranteed to be numerically stable.

To ensure the passivity of the equivalent circuit of Fig. 4.8, additional passive elements are introduced. At first glance, it appears that the passivity could be achieved by adding additional damping resistors across various elements in Fig. 4.8. However, one can verify that such a simple augmentation is insufficient for ensuring universal passivity i.e., for arbitrary values of the interfacing L and the time-step Δt.

Hence, to ensure universal passivity as seen from the circuit's driving ports 1 & 2, a mathematical construct is introduced by the adding extra elements (R_1, R_2 and C_2) in parallel as shown in the enclosed dashed boxes in Fig. 4.9. If the values for these elements are selected as in Table 4.1, then the circuit will be universally passive (this will be proved in the following Section 4.3.2). Note that other passivity guaranteeing topologies *may* be possible, but as will be clear in this chapter, the proposed one is shown (in Section 4.4, 4.5 & 4.6) to be adequately accurate.

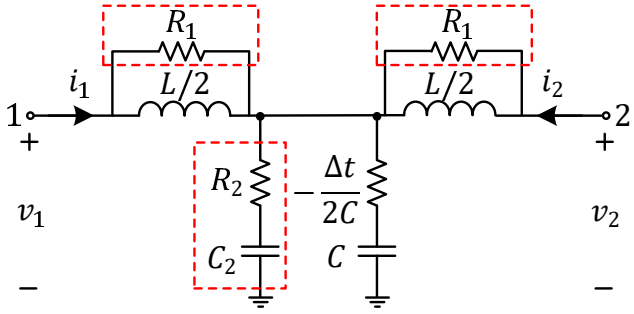


Figure 4.9: Universally Passive Network Partitioning (UPNP) interface

Table 4.1: Component Values for UPNP Interface of Fig. 4.9

Component	Value
C_2 $(= C)$	$(\Delta t)^2 / L$
R_1	$2L/\Delta t$
R_2	$L/\Delta t$

By applying the trapezoidal integration method to the interface circuit of Fig. 4.9, one obtains the companion circuit for the UPNP interface as in Fig. 4.10. As we can see, this

circuit can partition the system. For the circuit of Fig. 4.10, $V_h(t)$ can be calculated using (4.4).

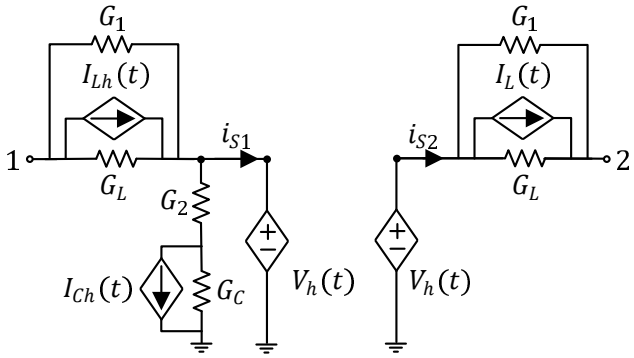


Figure 4.10: Partitioning using the UPNP Interface

$$V_h(t) = V_h(t - \Delta t) + \frac{\Delta t}{C}\left(i_{S1}(t - \Delta t) - i_{S2}(t - \Delta t)\right) \tag{4.4}$$

4.3.2 Proof for Universal Passivity of the UPNP Interface

To prove that the interface in Fig. 4.9 is universally passive as seen from the its driving ports 1 & 2 with R_1, R_2 and C_2 as in Table 4.1, we consider its two-port impedance matrix $\mathbf{Z}(s)$ and check whether it satisfies the following conditions for passivity [87].

Necessary and Sufficient Conditions for a Rational Function Matrix $\mathbf{Z}(s)$ to be Passive: [87]

1. Each element $z_{ij}(s)$ of $\mathbf{Z}(s)$ must be analytic in the open right half s-plane (i.e., $\Re\{s\} > 0$),

2. $\mathbf{Z}(j\omega) + \mathbf{Z}^H(j\omega) \succeq 0$ (i.e., is positive semidefinite) $\forall\ \omega \in \mathbb{R}$, except for simple poles $j\omega_0$ of $\mathbf{Z}(s)$, where the residue matrix must be positive semidefinite,

3. $\mathbf{Z}(-j\omega) = \mathbf{Z}^*(j\omega)$,

4. Asymptotically, $\mathbf{Z}(s) \to \mathbf{A}s$ in $\Re\{s\} > 0$, where $\mathbf{A}$ is a real, constant, symmetric positive semidefinite matrix.

The impedance matrix $\mathbf{Z}(s)$ for the UPNP interface of Fig. 4.9 is as in (4.5) with elements $z_{11}(s)$ and $z_{12}(s)$ as in (4.6) and (4.7) respectively. As the circuit is symmetrical, $z_{11}(s) = z_{22}(s)$ and because it is also reciprocal, $z_{12}(s) = z_{21}(s)$ [60].

$$\mathbf{Z}(s) = \begin{bmatrix} z_{11}(s) & z_{12}(s) \\ z_{12}(s) & z_{11}(s) \end{bmatrix} \tag{4.5}$$

$$z_{11}(s) = \frac{R_1 L s}{2R_1 + Ls} + \frac{\left(R_2 + \frac{1}{C_2 s}\right)\left(-\frac{\Delta t}{2C} + \frac{1}{Cs}\right)}{\left(R_2 - \frac{\Delta t}{2C}\right) + \left(\frac{1}{C_2 s} + \frac{1}{Cs}\right)} \tag{4.6}$$

$$z_{12}(s) = \frac{\left(R_2 + \frac{1}{C_2 s}\right)\left(-\frac{\Delta t}{2C} + \frac{1}{Cs}\right)}{\left(R_2 - \frac{\Delta t}{2C}\right) + \left(\frac{1}{C_2 s} + \frac{1}{Cs}\right)} \tag{4.7}$$

Substituting for C_2 ($= C$), R_1 and R_2 from Table 4.1, $z_{11}(s)$ and $z_{12}(s)$ are given by (4.8) and (4.9) in terms of L and Δt.

$$z_{11}(s) = \frac{2Ls}{4 + (\Delta t)\, s} + \frac{L\,(1 + (\Delta t)\, s)\,(-2 + (\Delta t)\, s)}{(\Delta t)^2\, s\,(4 + (\Delta t)\, s)} \tag{4.8}$$

$$z_{12}(s) = \frac{L\,(1 + (\Delta t)\, s)\,(-2 + (\Delta t)\, s)}{(\Delta t)^2\, s\,(4 + (\Delta t)\, s)} \tag{4.9}$$

From (4.8) and (4.9), we can conclude that $z_{11}(s)$ and $z_{12}(s)$ are analytic in $\Re\{s\} > 0$ since all the poles lie in the closed left half complex plane for all positive values of Δt. Hence, Condition 1 for passivity is satisfied. Similarly, it is easy to verify that Conditions 3 and 4 are also satisfied.

To check if Condition 2 is satisfied or not, we can derive the expressions for the eigenvalues of $\mathbf{Z}(j\omega) + \mathbf{Z}^H(j\omega)$ in terms of ω, Δt and L. For a matrix to be positive semidefinite, its eigenvalues should be non-negative [11]. The expressions for the eigenvalues of $\mathbf{Z}(j\omega) + \mathbf{Z}^H(j\omega)$ are given by (4.10). Derivation of (4.10) is given in Appendix E.

$$\lambda_1 = \frac{4\omega^2 L\Delta t}{(\Delta t)^2\omega^2 + 16}; \quad \lambda_2 = \frac{(8L/\Delta t)}{(\Delta t)^2\omega^2 + 16} \tag{4.10}$$

As $L > 0$ and $\Delta t > 0$, (4.10) clearly shows that the eigenvalues are non-negative. Hence, $\mathbf{Z}(j\omega) + \mathbf{Z}^H(j\omega) \succeq 0 \ \forall \ \omega \in \mathbb{R}$.

The above analysis proves the universal passivity of the proposed interface. Therefore, using the results in [65], we can conclude that the EMT simulation using the trapezoidal integration method will always be numerically stable if all other elements in the circuit are also passive. This applies even when power-electronic switches are present, as long as every switching state is passive [65].

4.4 Accuracy of the UPNP Interface

Both the stub-line and the UPNP interfaces replace an inductor with an interface that partitions the network. Hence, to evaluate the accuracy performance of these interfaces, in this section we compare the accuracy of simulation of these options with that of a circuit with the pure inductor. Firstly, for evaluating their accuracy over a wide frequency range, the frequency responses of their two-port admittance parameters are compared with that of an equivalent inductor. Next, for evaluating their accuracy at ac system frequency, we compare how the power flows in a typical network change when an inductor is replaced with an equivalent interface.

4.4.1 Comparison of Frequency Responses

A convenient metric for evaluating the accuracy of the UPNP as well as the stub-line interfaces over a wide range of frequencies is to see how well their frequency responses match with that of the original inductor. To do this, we consider the two-port admittance form for each of these approaches as given by (4.11). The parameters $y_{11}(j\omega)$ and $y_{12}(j\omega)$ in (4.11) take the forms as in (4.12) - (4.14) respectively for an inductor L, the proposed UPNP interface and the equivalent stub-line interface (of Section 4.2.3 - note (4.14) is obtained using the

circuit in Fig. 4.5 and (4.3)).

$$\mathbf{Y}(j\omega) = \begin{bmatrix} y_{11}(j\omega) & y_{12}(j\omega) \\ y_{12}(j\omega) & y_{11}(j\omega) \end{bmatrix} \tag{4.11}$$

Inductor:

$$y_{11}(j\omega) = \frac{1}{j\omega L};\;\; y_{12}(j\omega) = -\frac{1}{j\omega L} \tag{4.12}$$

UPNP Interface:

$$\begin{gathered} \mathbf{Y}(j\omega) = \mathbf{Z}(j\omega)^{-1} \\ \text{with } \mathbf{Z}(s) \text{ as in (4.5) with } z_{ij}(s) \text{ as in (4.8) and (4.9)} \end{gathered} \tag{4.13}$$

Stub-line Interface:

$$y_{11}(j\omega) = \left(\frac{\Delta t}{jL}\right)\frac{\cos(\omega\Delta t)}{\sin(\omega\Delta t)};\;\; y_{12}(j\omega) = -\frac{\Delta t}{jL\sin(\omega\Delta t)} \tag{4.14}$$

The comparison of the two-port admittance frequency responses of the inductor (i.e., no interface) with the stub-line interface and UPNP interface for the elements of $\mathbf{Y}(j\omega)$ are given in Fig. 4.11. In order to draw generalized conclusions applicable to arbitrary values of L or Δt, the frequency is per-unitized with a base of $1/(2\Delta t)$ and the admittance with a base of $\Delta t/(\pi L)$. Each response is plotted up to the Nyquist frequency $f = 1/(2\Delta t)$.

We can clearly see in Fig. 4.11 that the response of the UPNP interface tracks the inductor's response more closely compared to that of the stub-line interface. At low frequencies, the responses for both interfaces match the inductor's response closely. However, the stub-line exhibits sharp resonance peaks in the high-frequency range above $f = 1/(20\Delta t)$, which the UPNP interface does not.

Hence, we can conclude that the UPNP has better accuracy as it more closely resembles the behavior of an inductor over a wider frequency range compared to the stub-line interface.

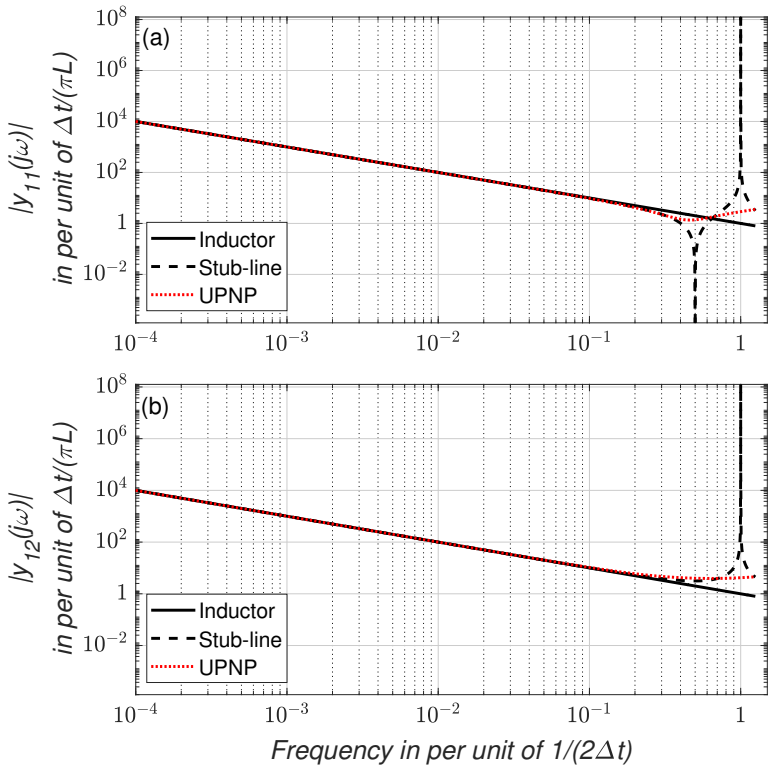


Figure 4.11: Frequency Response - UPNP interface, Stub-line interface & Inductor

4.4.2 Comparison of Power Flows at the Fundamental Frequency

Both the UPNP as well as the stub-line interfaces introduce extra 'fictitious' elements that aid in partitioning the system. However, when these interfaces are used to replace an inductor in a circuit, the fundamental frequency power flows at their terminals can change as compared to the base case of an inductor. Hence, for evaluating the accuracy of these interfaces at the fundamental frequency, we compare the change in the active as well as reactive power flows that are observed when an inductor is replaced with an equivalent UPNP or stub-line interface. The power flow changes are evaluated for different values of the inductor and for different simulation time-steps Δt.

For this, we consider a typical circuit shown in Fig. 4.12 that consists of a 100 MVA, 230/15 kV, 60 Hz two winding transformer with a leakage reactance X_t. The base case is the 'no interface' case, i.e., the circuit is simulated with the original leakage inductor. The

sending end voltage is taken as $1\angle 0$. The value of X_t is varied from 0.05 pu to 0.2 pu in steps of 0.01 pu. The receiving end voltage $V\angle\theta$ for each value of X_t is computed such that the active power P_S = 75 MW and the reactive power Q_S = 66.14 MVAr in steady-state.

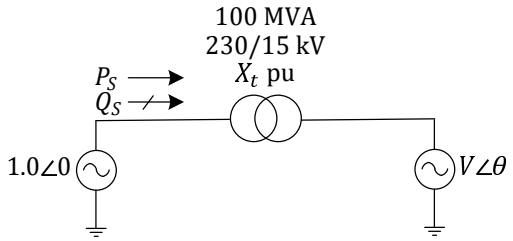


Figure 4.12: Circuit for comparison of change in steady-state power flows with delay-based interfaces

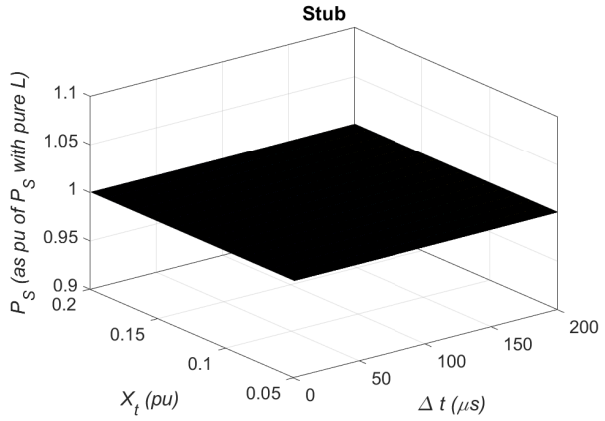
The following two cases are now considered:

- Case 1: X_t replaced with an equivalent stub-line interface.
- Case 2: X_t replaced with the proposed UPNP interface.

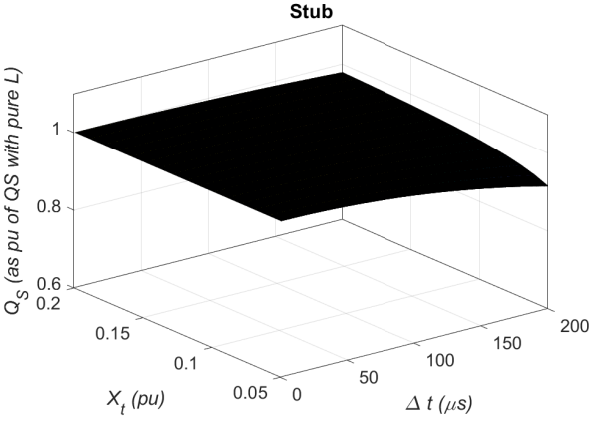
The three cases (viz. the base case, Case 1 and Case 2) are now compared when X_t is varied in the range [0.05 pu, 0.2 pu] and the time-step Δt in the range [1 μs, 200 μs]. Figs. 4.13a and 4.13b show the variation of P_S and Q_S as X_t and Δt are varied for the stub-line interface, while Figs. 4.13c and 4.13d show these for the proposed UPNP interface. Note that P_S in each case is per unitized with a base value of 75 MW, while Q_S in each case is per unitized with a base value of Q_S = 66.14 MVAr.

From these figures, it is apparent that at lower time-step values, the deviations for both cases compared to the base case of an inductor is negligible irrespective of the value of the interfacing X_t. However, as the time-step increases, the deviation also increases for both cases. The deviations for Case 2 (UPNP interface) are also observed to be marginally higher than Case 1 (stub-line case) when X_t is low and Δt is high. However, as shown in Table 4.2, for X_t = 0.1 pu (which is a typical value for the X_t of a transformer) and Δt = 50 μs (which is a typical value for Δt in EMT simulations), the deviations with the UPNP interface are 0.44 % for P_S and −1.05 % for Q_S, which are slightly higher (though still typically in

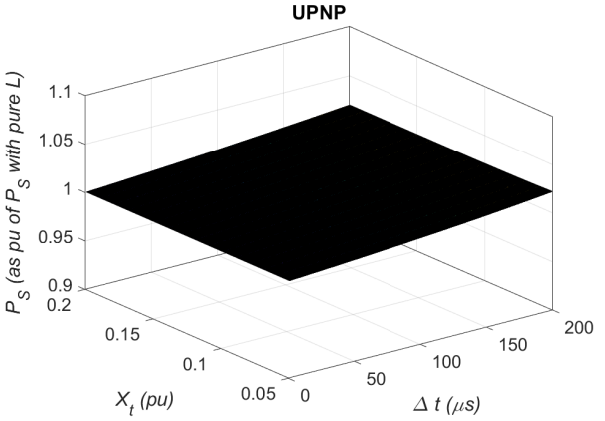
the 1 % range) than that with the stub line interface. Although, as will be demonstrated in Section 4.5, the UPNP has the advantage that it mitigates the high-frequency spurious oscillations problem that the stub-line interface manifests.



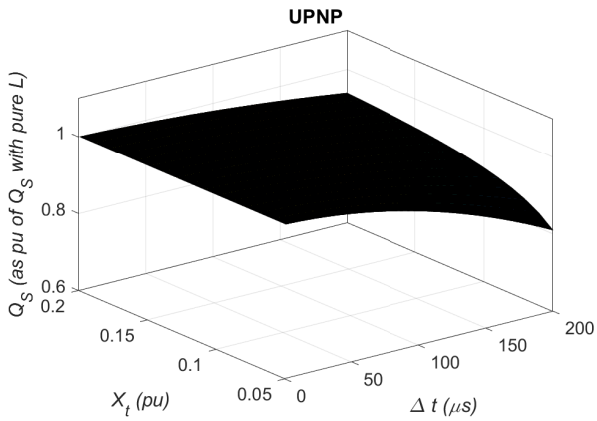
(a) P_S (Case 1 - Stub-line)



(b) Q_S (Case 1 - Stub-line)



(c) P_S (Case 2 - UPNP Interface)



(d) Q_S (Case 2 - UPNP Interface)

Figure 4.13: Change in steady-state power flows with delay-based interfaces in the circuit of Fig. 4.12

Table 4.2: P_S and Q_S for delay-based interfaces with $X_t = 0.1$ pu and $\Delta t = 50\ \mu s$ in the circuit of Fig. 4.12

	Stub-line Interface		UPNP Interface	
	Value	**% Deviation**	**Value**	**% Deviation**
P_S	1.0001 pu	0.01 %	1.0044 pu	0.44 %
Q_S	0.9974 pu	−0.26 %	0.9895 pu	−1.05 %

4.5 Ability of the UPNP Interface to Suppress Spurious High Frequency Oscillations

For a preliminary comparison of the UPNP interface with the stub-line interface, we consider the time-domain simulation of the circuit shown in Fig. 4.14.

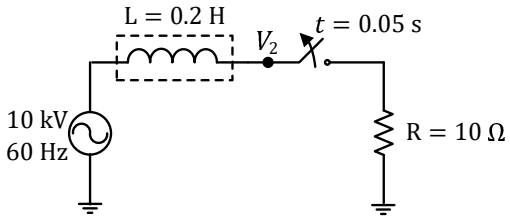


Figure 4.14: Circuit for preliminary comparison of delay-based interfaces

The following three cases are considered, all simulated using the trapezoidal method with time-step $\Delta t = 50\ \mu s$.

- Base Case: L modelled as an inductor (i.e., no interface)
- Case 1: L modelled using an equivalent stub-line interface.
- Case 2: L modelled using the proposed UPNP interface.

As is the case with most commercial simulation programs that use the trapezoidal integration method, a chatter removal method is also used in this implementation [75]. The switch is modelled as an ideal switch (i.e., $R_{on} = 0$ and $R_{off} \rightarrow \infty$) using the modified augmented nodal analysis (MANA) formulation [88]. The no-interface case (Case 1) of the inductor is used as reference for comparison with the interfaced Cases 1 and 2.

The switch is initially closed with the circuit operating in steady state and is opened at $t = 0.05\ s$. Fig. 4.15 presents a comparison of the simulation results for the voltage V_2 for the Base Case (inductor) and Case 1 (stub-line interface) while Fig. 4.16 presents the same comparison for the Base Case (inductor) and Case 2 (UPNP interface).

All the responses essentially overlap initially with the switch in closed position. After the switch opens, the stub-line interface shows sustained high frequency (HF) oscillations with

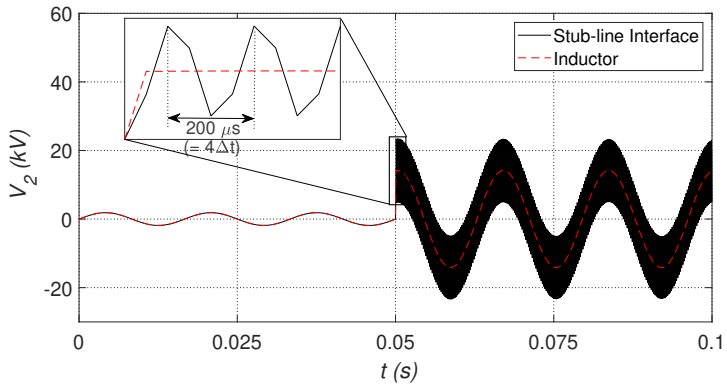


Figure 4.15: Response of voltage V_2 (Base Case – Inductor, and Case 1 – Stub-line interface)

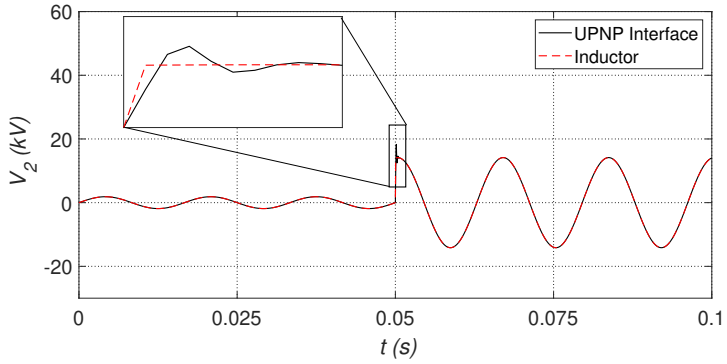


Figure 4.16: Response of voltage V_2 (Base Case – Inductor, and Case 2 – UPNP interface)

a period $4\Delta t$ (see Fig. 4.15) as compared to the response with an inductor. In contrast, we can clearly see in Fig. 4.16 that with the proposed UPNP interface, the oscillations, though present, are minimal and decay very rapidly. And the response with the UPNP interface matches very closely with that of an inductor even after the switch is opened.

Thus, we can see that the sustained high frequency oscillations are suppressed when the proposed UPNP interface is used instead of the stub-line interface for partitioning.

Note that the oscillation observed in the stub-line case is not the same as the phenomenon of chatter [89] which is observed when a switch in series with an inductor is opened in a circuit being simulated using the trapezoidal integration method. Chatter oscillation always has a period of $2\Delta t$ [89] as opposed to the $4\Delta t$ period observed here. Also, chatter oscillation

is easily suppressed by the method of [75] that is used here whereas the stub-line oscillation is not.

Reason for the spurious oscillations with stub-line interface:

The oscillations observed here are in fact because the inductor is represented by an equivalent stub-line in Case 1. The period $4\Delta t$ is due to the reflections of the wavefront at the open-circuit termination [90] after the switch is opened. It can be verified by plotting the admittance response of an open-circuited stub-line as shown in Fig. 4.17.

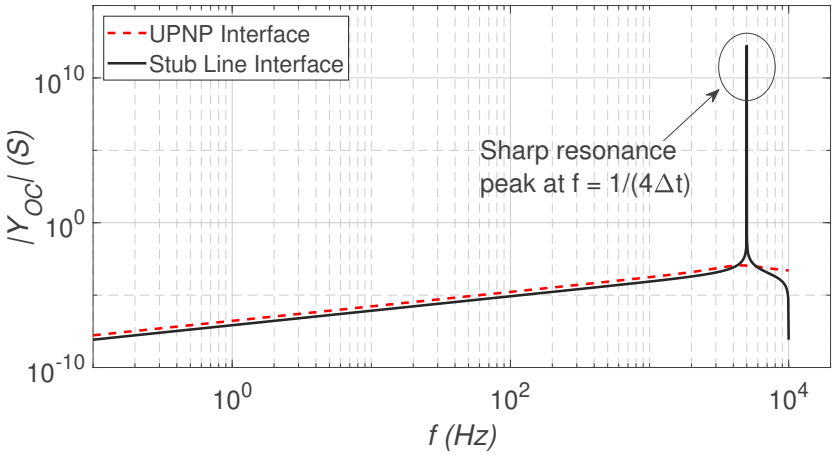


Figure 4.17: Open-circuit Admittance Response (Stub Line and UPNP Interface)

We can see that the stub-line has a sharp (undamped) resonance at $f = 1/(4\Delta t)$. In contrast, the open-circuit admittance response of the UPNP interface (also given Fig. 4.17) exhibits no sharp resonance peak and consequently does not result in sustained high frequency oscillations (as also seen from the time-domain simulation result in Fig. 4.16).

Advantages of the proposed UPNP interface over existing delay-based interfaces:

The UPNP interface as shown in Fig. 4.10, is straightforward to implement as the partitions are interfaced with each other using only voltage sources. The only information that needs to be transferred between the subsystems for calculating the value of these voltage sources are the history values of i_{S1} and i_{S2} as given by (4.4).

As discussed in previous sections, unlike the approach proposed in [3], the guaranteed

passivity of the UPNP interface ensures that it does not compromise the numerical stability of a simulation. Compared with the stub-line interface, it also has superior accuracy and suppressed spurious oscillatory behavior.

4.6 Simulations Examples

To demonstrate the practical utility of the UPNP interface, we consider three test cases with different types of HVdc converters which are simulated in PSCAD/EMTDC [71]. To check whether the UPNP interface is sufficiently accurate, it is compared with a reference simulation in which there is no partitioning, with the entire circuit modelled as one system.

4.6.1 Example 1 - Point-to-point VSC-HVdc Transmission System

Fig. 4.18 shows the test system for Example 1. It consists of a 200 MW, ±200 kV point-to-point voltage source converter (VSC) based HVdc link. The purpose of this example is to validate the accuracy the proposed UPNP interface with a more detailed circuit with power electronic converters.

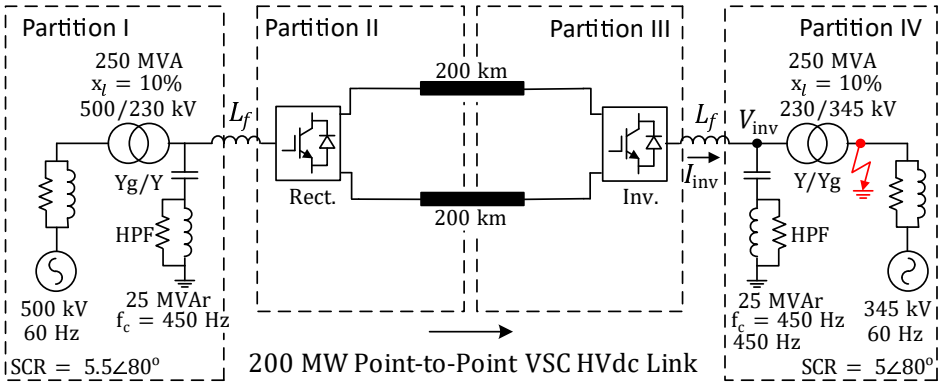
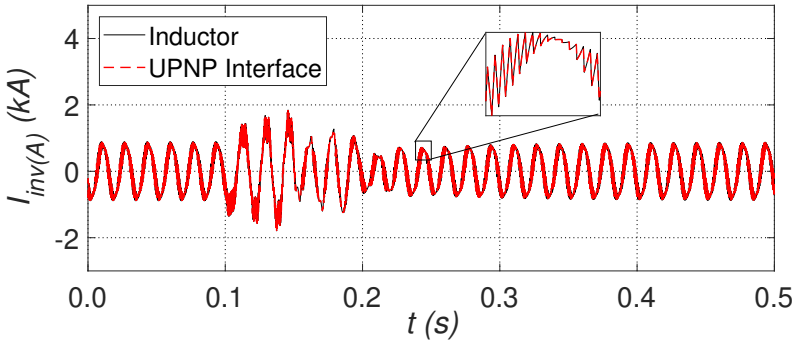


Figure 4.18: Example 1 – Point-to-point VSC-HVdc Link

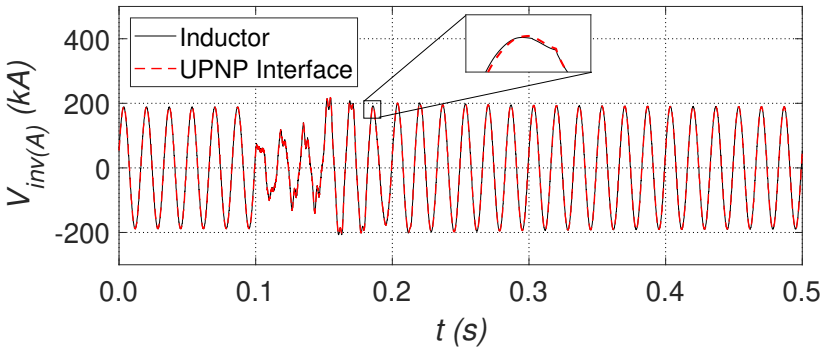
Both rectifier and inverter are two-level converters interconnected using two 200 km dc coaxial cables modelled using the frequency dependent universal line model [91]. The cable data is taken from [92]. Each converter is controlled using decoupled *dq*-control strategy [93]. The rectifier side controls the active power and its ac-bus voltage, while the inverter side

controls the dc voltage and its ac-bus voltage. The switching frequency for each converter is 1980 Hz (= 33 × 60 Hz).

The system is partitioned into four parts as shown in Fig. 4.18 using the natural delay of the dc cables and each converter inductor (L_f = 72.4 mH) replaced by the proposed UPNP interface. A fixed time-step of $\Delta t = 2\ \mu s$ is used for the simulation.



(a) Ac Side Current (Phase A) at Inverter End



(b) Ac Side Voltage (Phase A) at Inverter End

Figure 4.19: $I_{inv(A)}$ and $V_{inv(A)}$ in Example 1 (Inductor and UPNP Interface)

The system initially operates in steady state. At $t = 0.1\ s$, a solid phase-to-ground fault occurs at the location shown in Fig. 4.18 and is cleared after 50 ms. Fig. 4.19 shows the comparison of the simulation results for the case with the UPNP interface and that for the unpartitioned system with no interface (i.e., L_f modelled as an inductor).

Fig. 4.19a and 4.19b show the results for the ac side phase current $I_{inv(A)}$ and phase voltage $V_{inv(A)}$ for both the approaches. As we can see, the results match very closely not only during steady state but also after the fault occurs. The mean error in $I_{inv(A)}$ and $V_{inv(A)}$

is 0.002 kA and 0.2 kV respectively; or about 0.2% and 0.1% of their respective steady state peak values of 1 kA and 200 kV. Similarly, the results for the active power P_{inv} and reactive power Q_{inv} (shown in Fig. 4.20) at the inverter end also match very closely for both the approaches.

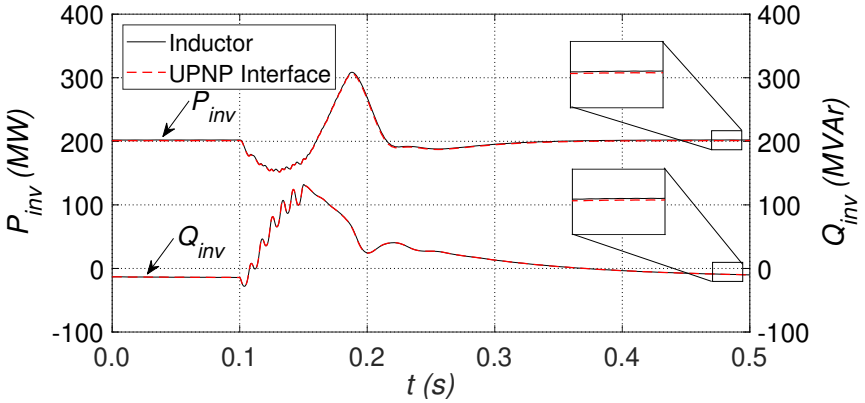


Figure 4.20: Active Power (P_{inv}) and Reactive Power (Q_{inv}) at Inverter End for Example 1 (Inductor and UPNP Interface)

Comparison between inductor, UPNP interface and stub-line:

The results shown in Fig. 4.19 and 4.20 only compared the responses of the case with the proposed UPNP interface with the reference case (i.e., with no interface).

Fig. 4.21 compares the startup transient waveforms of the reference case, a case with the stub-line interface and the proposed UPNP interface for the circuit in Fig. 4.18. With the stub-line interface, the system does not even settle down to a steady-state even after being allowed to run for about 5 s. In contrast, with all other parameters the same, the cases with the inductor and the proposed UPNP interface show essentially identical responses and rapidly settle down (in about 0.2 s) to the desired steady state operating point.

Hence, we can conclude from this example that the UPNP interface not only helps in partitioning, but also more closely matches the behavior of an inductor in comparison with an equivalent stub-line interface.

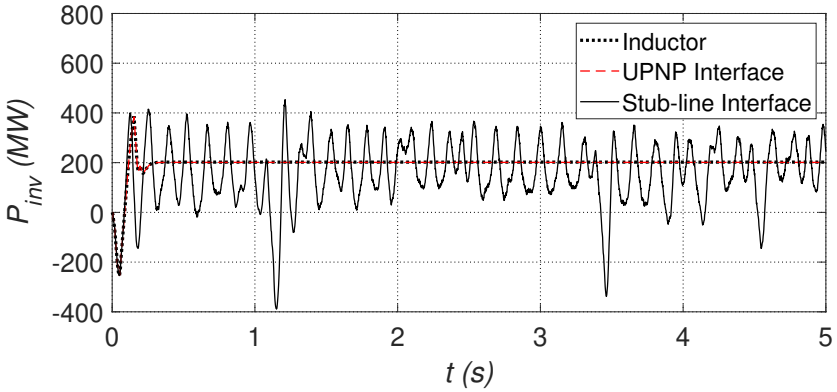


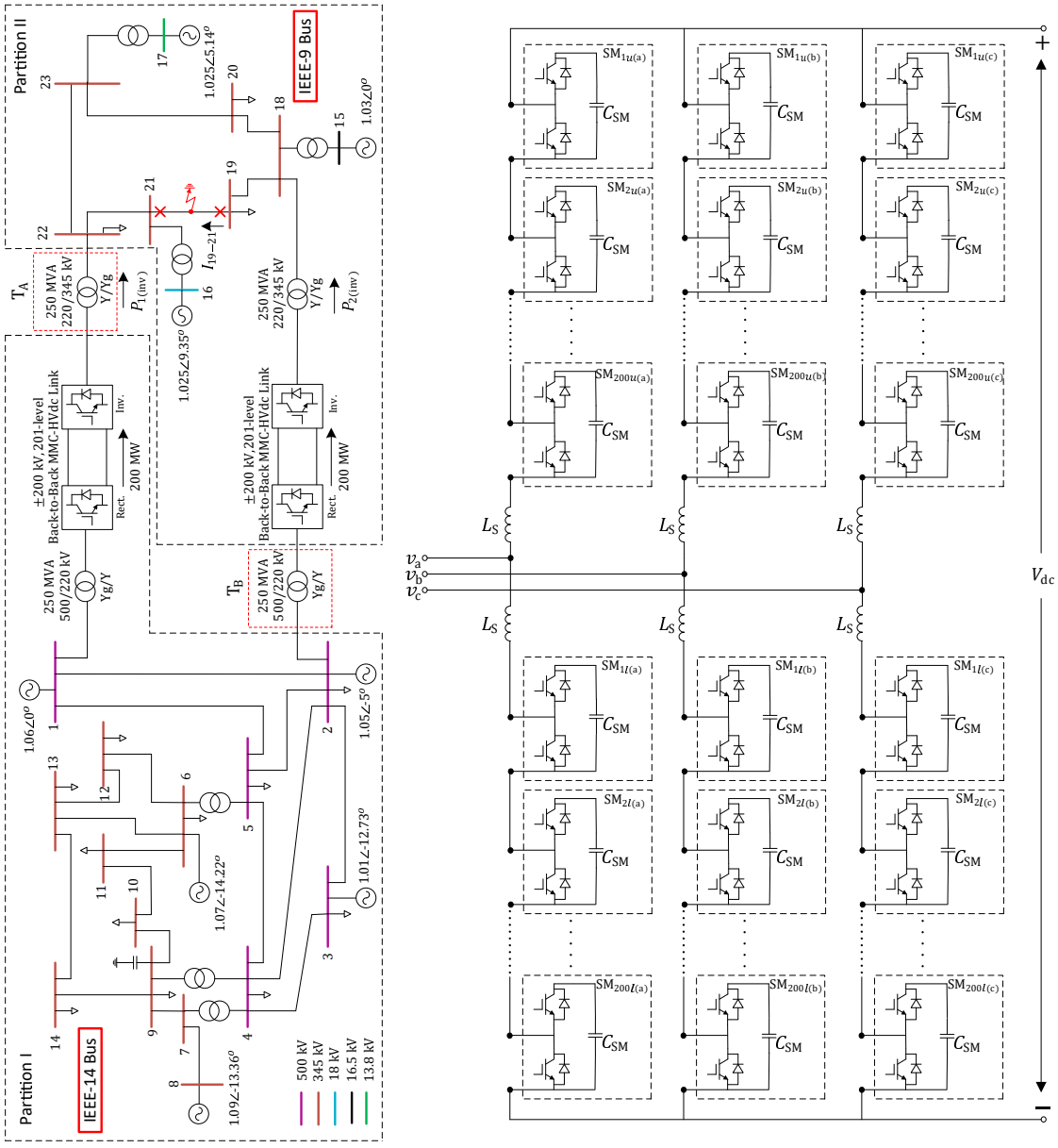
Figure 4.21: Active Power (P_{inv}) at Inverter End with Stub-line Interface

4.6.2 Example 2 - Two Asynchronous Areas interconnected using two Back-to-back MMC-HVdc Links

The second example system is as shown in Fig. 4.22. The purpose of this example is to verify the accuracy of the UPNP interface in a more complex system which has a topological resemblance with a real-world system [94] as well as to implement it on a parallel processing platform to demonstrate the achievable speedup.

The circuit comprises two asynchronous areas interconnected using two back-to-back modular multilevel converter (MMC) based HVdc links as shown in Fig. 4.22a. The topology is structurally similar to a real-world system reported in [94]. However, as the data for the actual system is not in the public domain, the sending and receiving end systems are modeled as the IEEE-14 bus and IEEE-9 bus systems respectively. The data for these is taken from [72].

Each MMC-HVdc link is rated at 200 MW, ±200 kV and has 201-level converters at each end consisting of half-bridge sub-modules (SM) as shown in Fig. 4.22b. The nominal SM voltage is 2 kV. Each arm inductor (L_S) for each converter is 0.15 pu and the converter transformer's leakage reactance is 0.18 pu [95]. Each SM capacitor (C_{SM}) is 2.5 mF which corresponds to a stored energy of 30 kJ/MW in the MMC [96]. Both the HVdc links use decoupled *dq*-control [93]. The rectifier side controls the active power and the reactive power at its ac-bus, while the inverter side controls the dc voltage and the reactive power at its



(a) Single Line Diagram **(b) MMC Converter**

Figure 4.22: Example 2 - Two asynchronous areas interconnected by two back-to-back MMC-HVdc Links

ac-bus. Nearest-level control (NLC) [97] is utilized for submodule switching.

To distribute the computational load more equitably, it is desirable to have each dc link simulated on a separate processor. Since both dc links are back-to-back, there are no transmission lines available to conveniently partition the system. Hence, the network is

partitioned into two parts as shown in Fig. 4.22. The ac system-side leakage inductance of the converter transformers T_A and T_B is used for partitioning using the proposed UPNP interface. Fig. 4.23 identifies the inductance used for the partitioning.

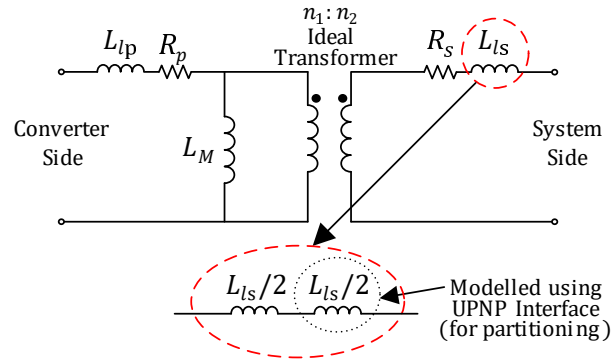


Figure 4.23: UPNP Interface-based partitioning using transformer's leakage

With the system initially operating in steady state, a solid three-phase fault is applied at the midpoint of line 19-21 at $t = 0.1$ s. It is cleared after 50 ms by opening the breakers at either ends of this line. Subsequently, the breakers are reclosed after a duration of 0.3 s. Fig. 4.24 and Fig. 4.25 compare the simulation results for the case with the UPNP interface, with the simulation results for the original unpartitioned system (i.e., with L_{ls} in Fig. 4.23 modelled as an inductor). The time-step used in this simulation is $\Delta t = 20$ μs.

Fig. 4.24 shows the current through the transmission line 19-21 for both the approaches. The results match very closely. The mean error in $I_{19\text{–}21(A)}$ is 0.0016 kA i.e., 0.8% based on the steady state peak value of about 0.2 kA. Similarly, the results for the inverter end active power P_{inv} (shown in Fig. 4.25) for both the links also match very closely for both the approaches. Thus, this verifies the accuracy of the UPNP interface.

A 5 s duration simulation was conducted with a time-step of $\Delta t = 20$ μs on a 2.1 GHz AMD Opteron 6272 sixteen core general-purpose Windows 10 PC with 64 GB RAM. For this case, the CPU execution times for the partitioned parallel simulation (with the UPNP interface) and the unpartitioned serial simulation are given in Table 4.3.

Table 4.3 shows that a simulation speedup of 2.14 is achieved in this case using parallel processing with the UPNP interface.

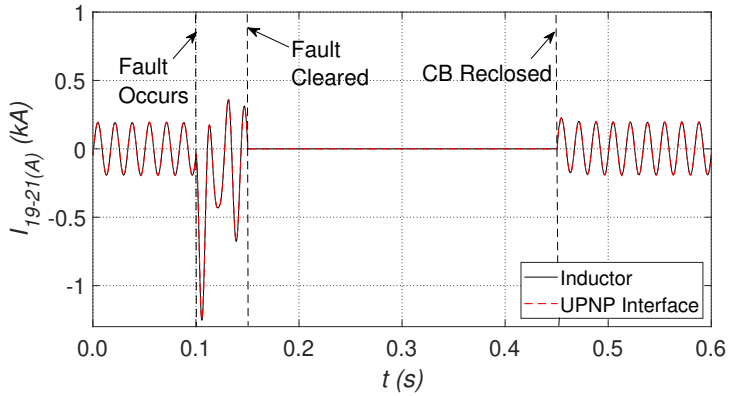
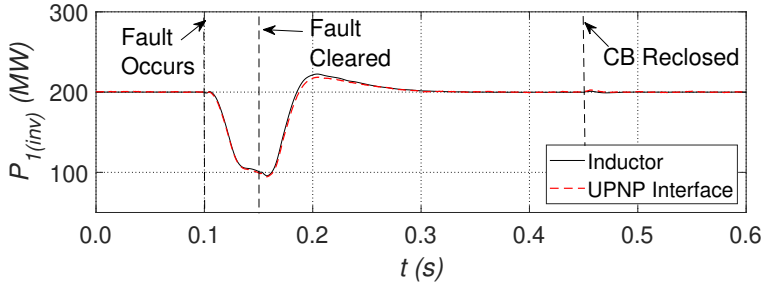
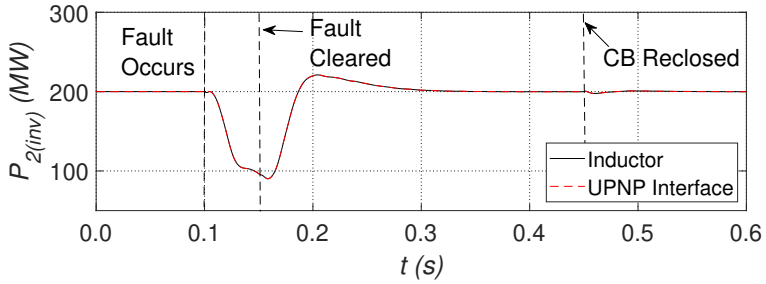


Figure 4.24: Line 19-21 current in Example 2 (Inductor and UPNP Interface)



(a) Link # 1



(b) Link # 2

Figure 4.25: Inverter active power in Example 2 (Inductor and UPNP Interface)

Table 4.3: Comparison of Total CPU Times (Example 2)

Serial Simulation T_S (sec)	Parallel Simulation T_P (sec)	Speedup (T_S/T_P)
1667.9	779.6	2.14

4.6.3 Example 3 - IEEE 39-bus System with an LCC-HVdc Link

In the third example, the standard IEEE 39-bus system is modified to include an embedded back-to-back line commutated converter (LCC) based HVdc link as shown in Fig. 4.26. The main purpose of this example is to verify the accuracy of the UPNP interface when it is used along with another delay-based interface (i.e., a transmission line) in a larger system, and also to implement it on a parallel processing platform to demonstrate the achievable speedup.

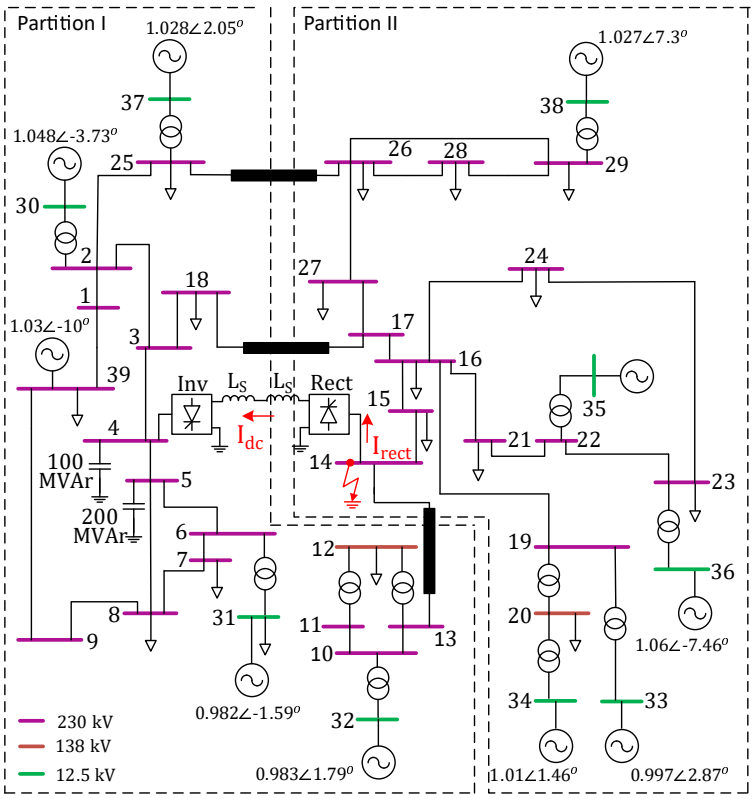
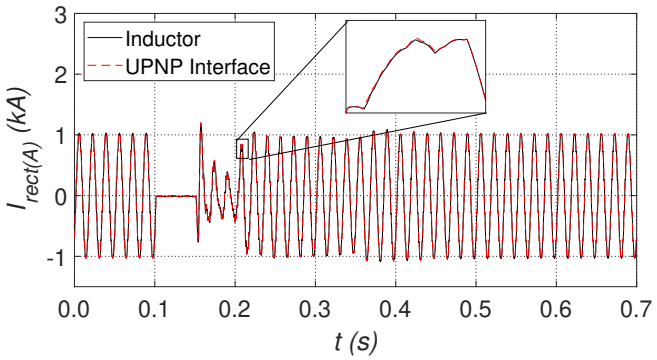


Figure 4.26: Example 3 – IEEE 39 Bus System with LCC-HVdc Link

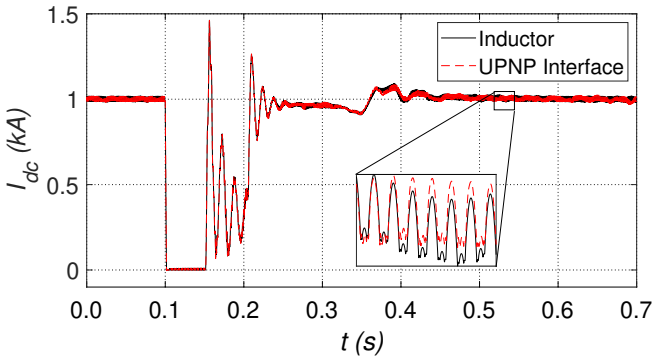
Appendix A shows the configuration and component values for the LCC-HVdc link which interconnects Bus 14 and Bus 4. The rectifier and inverter are twelve-pulse converters with the controls same as in [73]. The data for the IEEE-39 bus system is taken from [72].

The system is partitioned as shown in Fig. 4.26. Since the LCC-HVdc system is a back-to-back scheme, there is no transmission line connecting the two converters to allow

for partitioning. Hence the LCC-HVdc converter's 34 mH dc-side smoothing reactor (L_S) [98] is utilized to partition the system using the proposed UPNP interface (see Fig. A.1 in Appendix A for details). The value for L_S corresponds to that used in a real-world back-to-back system [99]. In addition to the UPNP interface-based partitioning, transmission lines 25–26, 17–18 and 13–14 are modelled using the distributed parameter model to allow for partitioning the system. A simulation time-step of Δt = 20 μs is used. The system initially operates in steady state. At t = 0.1 s, a solid three-phase-to-ground fault occurs near the rectifier bus which is cleared after 50 ms.



(a) Phase A current on rectifier's ac side



(b) Results for Example 3 (Inductor and UPNP Interface)

Figure 4.27: Results for Example 3 (Inductor and UPNP Interface)

Fig. 4.27 show the comparison of the simulation results for the case with UPNP approach and that for the unpartitioned system with no interface (i.e., both L_S modelled just as inductors). As we can see in Fig. 4.27, the results match very closely. The mean error in

$I_{rect(A)}$ is 0.002 kA which is 0.2 % based on its steady state peak value of 1 kA. Similarly, the results for I_{dc} also match closely. Thus, this verifies the accuracy of the UPNP interface.

For a 5 s simulation with Δt = 20 μs, the CPU execution times for the partitioned parallel simulation (with the UPNP interface) and the unpartitioned serial simulation are given in Table 4.4 (these were measured on the same PC as for Example 2). It is evident from Table 4.4 that partitioning the system into two roughly equally sized subsystems resulted in a speedup of 2.27.

Table 4.4: Comparison of Total CPU Times (Example 3)

Serial Simulation T_S (sec)	**Parallel Simulation T_P (sec)**	**Speedup (T_S/T_P)**
67.81	29.95	2.27

Thus, the previous two examples demonstrate that the UPNP interface facilitates implementation on a parallel computing platform without compromising the numerical stability of the simulation.

4.7 Conclusions

Transmission lines have long been used to partition the network into subsystems suitable for parallel processing implementations. However, they may not always be available at convenient partitioning locations. To address this limitation, in this chapter we develop a novel delay-based interface which permits partitioning of the system at branches where series inductors are present. As series inductors are widespread in a power network, this greatly increases the number of partitioning opportunities. Since the proposed method always generates a strictly passive interface, it does not contribute negatively to the stability of the simulation.

The resulting sub-network partitions are interfaced with each other in a straightforward manner using voltage sources that depend only on history values and permit speed-up of the simulation by parallelizing the network solution. The work reported in this chapter

uses time-domain simulations backed up by the per-unitized frequency response plots of the admittance matrix elements of the interface to show that it has better accuracy compared to the widely used stub-line interface.

Simulation test cases with HVdc converters are used to demonstrate the practical utility of this interface. It is also demonstrated that the proposed approach facilitates implementation on a parallel computing platform without compromising the numerical stability of the simulation.

4.8 Contributions

1. Developed a novel delay-based interface (called the Universally Passive Network Partitioning (UPNP) interface) that uses existing inductors in the circuit to partition the network, thus allowing for simulation speed up when the solution of the partitioned network is computed on a parallel computing platform.

2. Mathematically proved that the proposed interface is guaranteed to be passive for any arbitrary value of the inductor (L) and time-step (Δt).

3. Analyzed the accuracy performance of the proposed interface to show that it exhibits superior performance compared to the existing delay-based partitoning methods.

4. Demonstrated the practical utility of the UPNP interface using simulation test cases that contained HVdc converters.

Chapter 5

Stability Assessment of Multi-rate EMT Simulations of LTI Circuits

This chapter presents a novel approach for assessing the numerical stability of multi-rate EMT simulations of LTI circuits. It is shown that such simulations always yield a periodically varying system in the discrete-time domain. By exploiting this property and applying the well-known 'lifting' technique, a sampled data time-invariant representation is obtained for the simulated system which is then used for stability assessment using eigenvalue analysis.

5.1 Introduction

In Chapters 2 - 4, we discussed various methods for accelerating single-rate Electromagnetic Transient (EMT) simulations using parallel processing. Single-rate EMT simulations use a single fixed time-step Δt for discretizing the equations of all the components in a network. Power systems usually consist of a combination of "fast" and "slow" components that require widely different time-steps for accurately modelling their transient behavior in a simulation [50]. If a single time-step Δt is used for discretizing all the components, there is a potential for the simulation to become excessively slow and computationally expensive. This is because this time-step Δt, which is selected based on the accuracy requirements of

the fastest component in a system, is also used for solving relatively slower portions of the system.

One of the alternatives that is proposed in the literature is to divide the system into multiple subsystems and then use distinct time-steps for simulating each subsystem. This is essentially the *multi-rate simulation* approach. The time-steps for each subsystem are selected based on their individual time response characteristics [51] and the solution for the entire system is reconciled every time the simulation time grid for the various subsystems coincides with each other. Multi-rate simulation techniques are particularly useful in real-time EMT simulators for enhancing the simulator's computational efficiency. Several multi-rate simulation methods have been proposed in literature - these include iterative approaches [53, 100] and non-iterative approaches [2, 50, 54]. The former type uses iterations when reconciling the solution for the entire system while the latter type uses direct methods which avoid any type of iterations. As real-time simulators must always satisfy the *real-time constraint* (i.e., its output must be synchronized with the real-world clock), non-iterative approaches are generally the preferred choice.

Most of the existing literature [2,50,54] on multi-rate simulation methods has focused on demonstrating two things: (i) the accuracy of their respective approaches; (ii) the enhancement of computational efficiency accrued from using such approaches in EMT simulations. However, very little attention has been directed toward the development of techniques for assessing the numerical stability of such simulations. Reference [101] does try to address this issue. However, it uses a linearized system for stability assessment which is inherently an approximation. Additionally, the theory behind their proposed stability assessment method is established using a standard state-space formulation by partitioning the state vector into slow and fast states. However, such a formulation does not entertain the widely used multi-rate EMT simulation methods which typically use network partitioning and not state vector partitioning (e.g., [2, 50]).

In this chapter, we develop a novel approach for assessing the stability of non-iterative multi-rate EMT simulations of linear time-invariant (LTI) circuits. The effectiveness of the

proposed approach is demonstrated by showing how it can accurately predict the stability of a commonly used multi-rate simulation method proposed in [2].

Firstly, it is demonstrated that multi-rate EMT simulations can produce unstable results for stable continuous-time LTI circuits *even when the trapezoidal integration method is used for discretizing various components in each subsystem.* Further, it is shown that such simulations always yield a periodically varying system in the discrete-time domain. By exploiting this property and using the well-known technique of 'lifting' [56], a sampled data time-invariant representation is obtained for the simulated discrete-time system. This is then used for assessing the numerical stability of the simulation using eigenvalue analysis. In this work, the dynamical equations for the simulated discrete-time system are formulated by adapting the procedure given in [1] for obtaining a discrete-time state space representation of a network directly from its companion circuit formulation.

It should be noted that the multi-rate simulation method of [2] is merely used to exemplify the effectiveness of the proposed approach in predicting stability. The approach itself is entirely general and can be applied to the stability assessment of any arbitrary non-iterative multi-rate simulation algorithm.

5.2 Key Differences between Single-rate and Multi-rate EMT Simulations

Single-rate EMT simulations use a single fixed time-step for discretizing the equations of all the components in a network. On the other hand, multi-rate EMT simulations use distinct time-steps for discretizing different portions of a circuit (e.g., in the illustration of a dual-rate case in Fig. 5.1, a large time-step ΔT is used for the slow subsystem while a small time-step Δt is used for the fast subsystem, where $\Delta T > \Delta t$).

Additionally, while exchanging the values of the interface variables between the fast and the slow subsystems, up-sampling and down-sampling are also needed in multi-rate EMT simulations because of the distinct time-steps on the two sides. Typical up-sampling methods

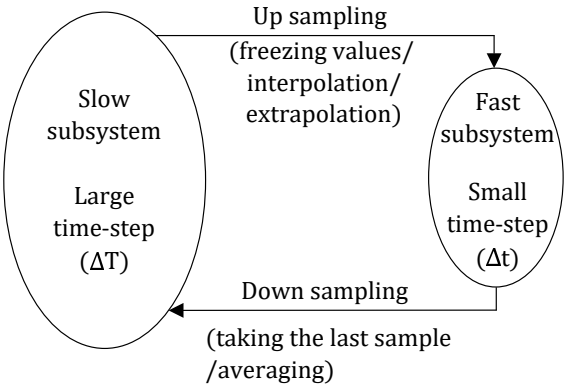


Figure 5.1: General Structure of Multi-rate EMT Simulations

include freezing of the slow variables at the interface [51] or interpolation/extrapolation using the previous two samples of the slow variables at the interface [2, 50, 54]. On the other hand, typical down-sampling methods include taking the last sample of the fast variables at the interface [51] or averaging the fast variables at the interface over the previous large time-step interval [2].

5.3 Motivation

This research on stability assessment of multi-rate EMT simulations is motivated by a case where an LTI network (which is stable in the continuous time-domain) is simulated as a multi-rate simulation using the trapezoidal integration method in each subsystem and the simulation, somewhat surprisingly, becomes numerically unstable. This case is discussed below.

Example to Demonstrate Instability of Multi-rate EMT Simulations

Consider the circuit shown in Fig. 5.2. Firstly, it is clear that this circuit is a stable LTI system. Hence, its single-rate simulation using the trapezoidal integration method is guaranteed to be numerically stable for any arbitrary value of the time-step [102]. The waveforms for the voltage at node 3 shown in Fig. 5.3 for $\Delta T = \Delta t = 5\ \mu s$ and $\Delta T = \Delta t = 50\ \mu s$ respectively are both stable and thus corroborate this fact.

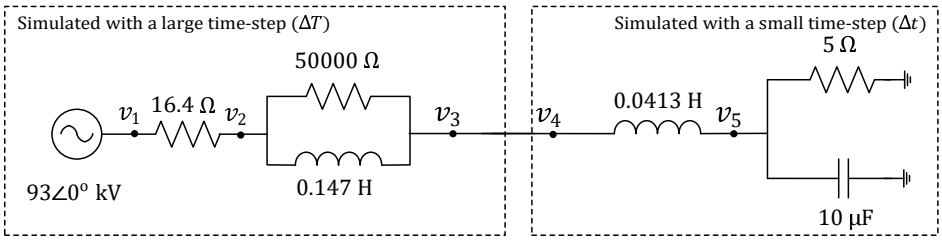
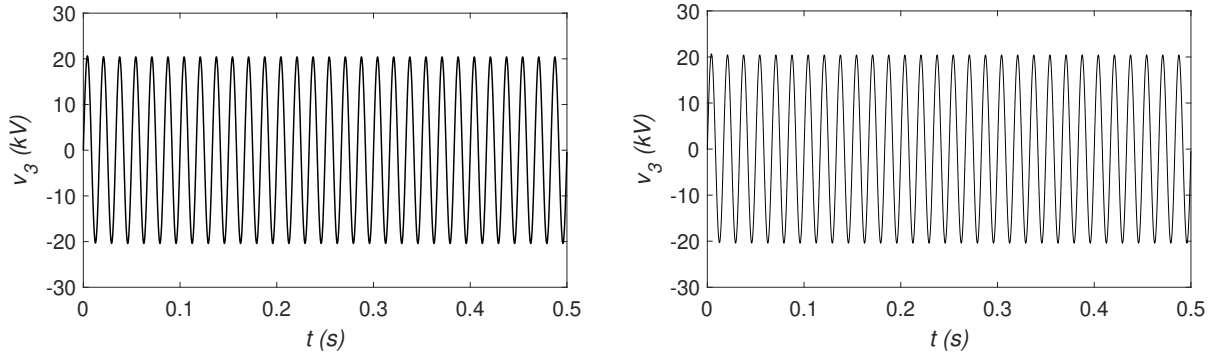


Figure 5.2: Example to Demonstrate the Instability of Multi-rate Simulations of Stable Continuous Time LTI Circuits



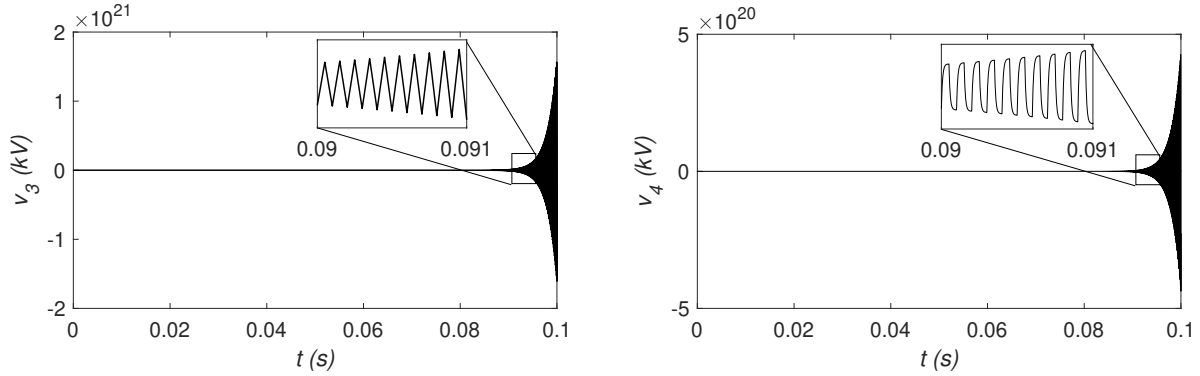
(a) With $\Delta T = \Delta t = 5\ \mu s$ **(b) With $\Delta T = \Delta t = 50\ \mu s$**

Figure 5.3: Voltage at Node 3 for the Circuit of Fig. 5.2 (Single-rate case, i.e., $\Delta T = \Delta t$)

The circuit in Fig. 5.2 is now simulated using the multi-rate simulation method of [2] with a large time-step $\Delta T = 50\ \mu s$ for the left side sub-network and a small time-step $\Delta t = 5\ \mu s$ for the right side sub-network. The multi-rate simulation method in [2] uses the trapezoidal integration method for discretizing the dynamical elements in each subsystem and uses the Multi Area Thevenin Equivalent (MATE) concept for partitioning and reconciling the solution of the entire system. The interface node is numbered "3" on the left side and "4" on the right side in Fig. 5.2. Fig. 5.4 shows the waveforms of these node voltages on either side of the interface.

We see that in the multi-rate case, the simulation is numerically unstable with this choice of time-steps. Thus, this example clearly shows that the property of guaranteed stability afforded by the trapezoidal integration method for LTI circuits in single-rate simulations does not generally guarantee the stability in multi-rate simulations. This example also points out

the need for developing a specialized method for the stability assessment of multi-rate EMT simulations. In the subsequent sections, we will develop a novel approach for the stability assessment of multi-rate EMT simulations of LTI circuits.



(a) Node Voltage v_3 (from ΔT side) **(b) Node Voltage v_4 (from Δt side)**

Figure 5.4: Voltages at Node 3 and 4 for the Circuit of Fig. 5.2 (Multi-rate Case)

5.4 Theoretical Framework for the Stability Assessment Method

We will now develop the mathematical framework for the method that is used for the stability assessment of multi-rate simulations of autonomous LTI circuits, i.e., ones without any external excitation. The reader is reminded that the exponential stability of an autonomous system implies Bounded Input Bounded Output (BIBO) stability of its non-autonomous counterpart [103]. Hence any conclusions about the stability that are derived for the autonomous case become directly applicable to the equivalent non-autonomous case.

5.4.1 State-space Model of Autonomous Linear Systems

For an autonomous linear system, the state-space model in the continuous time-domain can be written as in (5.1).

$$\underline{\dot{x}} = \mathbf{A}\,\underline{x} \tag{5.1}$$

By using any numerical integration method with a time-step Δt, the state vector in the current time-step (say $\underline{x}(t+\Delta t)$) can be related to the state vector in the previous time-step ($\underline{x}(t)$) using a linear transformation as in (5.2).

$$\underline{x}(t+\Delta t) = \mathbf{G}\,\underline{x}(t) \tag{5.2}$$

For example, $\mathbf{G} = \left(\mathbf{I} - \frac{\mathbf{A}\Delta t}{2}\right)^{-1}\left(\mathbf{I} + \frac{\mathbf{A}\Delta t}{2}\right)$ with the trapezoidal integration method whereas $\mathbf{G} = (\mathbf{I} + \mathbf{A}\Delta t)$ with the rectangular integration (or Forward Euler) method.

5.4.2 Periodicity in the Multi-rate Simulations of LTI Circuits

Fig. 5.5 shows a typical timeline of multi-rate simulations. Here, ΔT is the large time-step (used in the slow subsystem) whereas Δt is the small time-step (used in the fast subsystem). ΔT is assumed to be an integer multiple of Δt, i.e., $\Delta T = N\Delta t$ where N is a positive integer greater than 1.

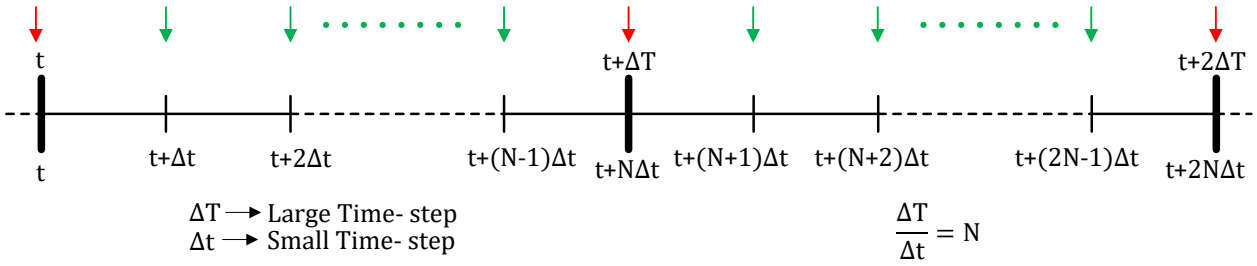


Figure 5.5: Timeline of Multi-rate Simulations

The following general steps are followed in each ΔT interval in any multi-rate simulation method:

(a) At every intermediate Δt instant within ΔT (indicated by green arrows in Fig. 5.5), the states of only the fast subsystem are updated; the slow subsystem's states remain latent at these intermediate instants.

(b) At every time instant which is an integer multiple of ΔT (indicated by red arrows in Fig. 5.5), the states of both the fast and the slow subsystem are updated.

Thus, starting from t with the initial conditions $\underline{x}(t)$, the difference equations for updating the states $\underline{x}$ in the first ΔT interval in Fig. 5.5 can be written as in (5.3).

$$\begin{aligned}
\underline{x}(t+\Delta t) &= \mathbf{G}_0\, \underline{x}(t) \\
\underline{x}(t+2\Delta t) &= \mathbf{G}_1\, \underline{x}(t+\Delta t) \\
&\vdots \\
\underline{x}(t+(N-1)\Delta t) &= \mathbf{G}_{N-2}\, \underline{x}(t+(N-2)\Delta t) \\
\underline{x}(t+N\Delta t) &= \mathbf{G}_{N-1}\, \underline{x}(t+(N-1)\Delta t)
\end{aligned} \tag{5.3}$$

Because of the selective updating of the fast and slow states within a ΔT interval, the matrices $\mathbf{G}_0$, $\mathbf{G}_1$, ... $\mathbf{G}_{N-1}$ would typically be distinct from each other. However, as the simulated circuit is LTI in the continuous time-domain, we can say that the same $\mathbf{G}_j$ ($j = 0, 1, \ldots N-1$) matrices are applicable for every subsequent ΔT interval. This is illustrated in Fig. 5.6.

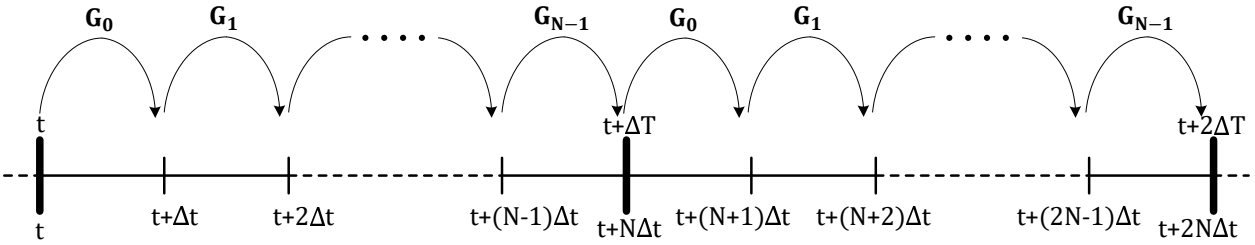


Figure 5.6: Periodic Nature of Multi-rate Simulations of LTI Circuits

Therefore, we can say that for a continuous-time LTI circuit, any multi-rate simulation method yields a periodically varying system in the discrete time-domain with the period equal to the large time-step ΔT and the overall transition matrix $\mathbb{G}$ as in (5.4).

$$\mathbb{G} = \mathbf{G}_{N-1} \times \mathbf{G}_{N-2} \times \ldots \mathbf{G}_1 \times \mathbf{G}_0 \tag{5.4}$$

5.4.3 Obtaining a Time-invariant Representation for a Periodically Varying Discrete Time System

For any periodically varying discrete-time system (such as the one in Fig. 5.6), it is possible to obtain a time-invariant representation by applying the 'lifting' technique [56]. In this technique, a lifting operator L is defined that maps $\underline{v} \longmapsto \underline{V}$ where $\underline{v}$ is a vector sampled at the regular sampling rate (Δt in case of Fig. 5.6) whereas $\underline{V}$ consists of N consecutive samples of $\underline{v}$ (where $N\Delta t = \Delta T$ being the period). Lifting the state vector $\underline{x}$ for the periodically varying discrete time system shown in Fig. 5.6 yields the following:

$$\underline{X}(t) = \begin{bmatrix} \underline{x}(t) \\ \underline{x}(t+\Delta t) \\ \vdots \\ \underline{x}(t+(N-2)\Delta t) \\ \underline{x}(t+(N-1)\Delta t) \end{bmatrix}; \quad \underline{X}(t+\Delta T) = \begin{bmatrix} \underline{x}(t+N\Delta t) \\ \underline{x}(t+(N+1)\Delta t) \\ \vdots \\ \underline{x}(t+(2N-2)\Delta t) \\ \underline{x}(t+(2N-1)\Delta t) \end{bmatrix} \tag{5.5}$$

Note that with this procedure, every 'lifted' vector $\underline{X}(t)$ includes each $\underline{x}(t+k\Delta t)$, where $k = 0, 1, \ldots N-1$ within a periodic interval ΔT, as can be seen in (5.5). Using the lifted state vectors in (5.5) and the difference equations in (5.3), we can obtain a time-invariant representation as in (5.6).

$$\mathbf{M}\,\underline{X}(t+\Delta T) = \mathbf{H}\,\underline{X}(t) \tag{5.6}$$

where, $\mathbf{M}$ and $\mathbf{H}$ matrices are as in (5.7).

$$\mathbf{M} = \begin{bmatrix} \mathbf{I} & & & & \\ & \mathbf{0} & & & \\ & & \ddots & & \\ & & & \mathbf{0} & \\ & & & & \mathbf{0} \end{bmatrix}; \qquad \mathbf{H} = \begin{bmatrix} & & & & \mathbf{G}_{N-1} \\ & & & \mathbf{G}_{N-2} & -\mathbf{I} \\ & & \iddots & \iddots & \\ & \mathbf{G}_1 & -\mathbf{I} & & \\ \mathbf{G}_0 & -\mathbf{I} & & & \end{bmatrix} \tag{5.7}$$

In (5.7), all entries which are blank are zeros, $\mathbf{I}$ is an identity matrix of appropriate size, $\mathbf{0}$ is a null matrix of appropriate size and $\mathbf{G}_0$, $\mathbf{G}_1$, ... $\mathbf{G}_{N-1}$ are matrices from (5.3).

Using the time-invariant model in (5.6), we can say that the periodically varying discrete-time system is stable if all the finite eigenvalues of the matrix pencil $(\mathbf{H}, \mathbf{M})$ are inside the unit circle.

NOTE: It can be shown that $\mathbf{M}$ in (5.6) will always be rank deficient for $N > 1$. Hence, in addition to the finite eigenvalues, the matrix pencil $(\mathbf{H}, \mathbf{M})$ will have additional eigenvalues of magnitude ∞. These eigenvalues are ignored [104] as they do not affect the system dynamics.

5.5 Formulation of Equations for Stability Assessment of Multi-rate EMT Simulations of LTI Circuits

Based on the discussion in the previous section, the following procedure can be adopted for obtaining a time-invariant representation for any arbitrary multi-rate simulation algorithm:

1. Within the ΔT interval, write down the governing equations for each Δt instant (similar to what was done in (5.3)).

2. Using these equations, obtain a time-invariant representation (similar to (5.6)).

The challenge now lies in writing down the governing equations for any arbitrary multi-rate simulation method. The mathematical framework developed in the previous section used a standard state-space representation. Most commonly used multi-rate EMT simulation approaches use Dommel's method [9] for formulating the equations of a network. Reference [63] has analytically proven that these two approaches are equivalent.

However, most multi-rate simulation methods proposed in the literature (e.g., [2, 50, 55]) also use network partitioning in order to yield multiple sub-networks to be simulated with distinct time-steps. Writing down the equations for such methods in a standard state-space

form can be onerous, typically requiring a lot of intermediate matrix manipulations. To circumvent this issue, in this work we adapt and use the approach of *Hollman and Marti* [1] for obtaining a discrete-time state-space representation directly from Dommel's formulation.

In this section, we discuss the detailed procedure for formulating the equations for the stability assessment of multi-rate EMT simulations of LTI circuits. The MATE-based method [2] is used as a demonstration example. However, it should be noted that the approach itself is entirely general and can be applied to the stability assessment of any arbitrary non-iterative multi-rate simulation algorithm.

5.5.1 Obtaining a State-space Representation from the Companion Circuit of a Network [1]

In this section, we briefly review the approach of [1] that is useful for obtaining a discrete-time state-space representation directly from Dommel's formulation. We consider an autonomous case, i.e., a circuit with no external sources.

The companion circuit representation of an inductor and capacitor at time t is shown in Fig. 5.7. If we know the network's solution at time t, then the history current source h_L for an inductor and h_C for a capacitor at the time $(t + \Delta t)$ are as in (5.8).

$$\begin{aligned} h_L(t + \Delta t) &= i_L(t) + g_L v_L(t) \\ h_C(t + \Delta t) &= -i_C(t) - g_C v_C(t) \end{aligned} \tag{5.8}$$

Here, $g_L = \Delta t/2L$, and $g_C = 2C/\Delta t$ with Δt being the simulation time-step. Using the companion circuit representation in Fig. 5.7, we can eliminate $i_L(t)$ and $i_C(t)$ from the expressions in (5.8) to yield (5.9).

$$\begin{aligned} h_L(t + \Delta t) &= h_L(t) + 2g_L v_L(t) \\ h_C(t + \Delta t) &= -h_C(t) - 2g_C v_C(t) \end{aligned} \tag{5.9}$$

Now knowing these relations for each inductor and capacitor, the expressions for the his-

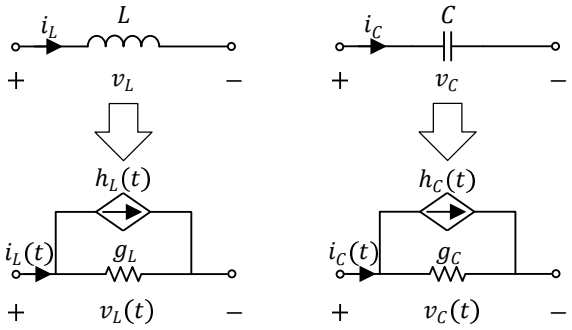


Figure 5.7: Inductor and capacitor, and their corresponding companion circuits

tory currents of all branch inductors and capacitors in any arbitrary circuit (containing n_L inductors and n_C capacitors) can be compactly written as in (5.10).

$$\underline{h}_b(t + \Delta t) = \mathbf{K}\, \underline{h}_b(t) + \mathbf{Y_b}\, \underline{v}_b(t) \tag{5.10}$$

Here,

$\underline{h}_b(t)$: Vector containing history currents for each inductor and capacitor.

$$= \begin{bmatrix} h_{L(1)} & \dots & h_{L(n_L)} & h_{C(1)} & \dots & h_{C(n_C)} \end{bmatrix}^T$$

$\mathbf{K}$: A diagonal matrix with entries equal to either 1 or −1.

$\mathbf{K}(i,i) = 1$ if i^{th} element in $\underline{h}_b(t)$ is an inductor.

$= -1$ if i^{th} element in $\underline{h}_b(t)$ is an capacitor.

$\underline{v}_b(t)$: Vector containing branch voltages for each inductor and capacitor.

$\mathbf{Y_b}$: Diagonal matrix of companion circuit conductance values for each inductor and capacitor (as given in (5.11)).

$$\mathbf{Y_b} = \begin{bmatrix} 2g_{L(1)} & & & & & \\ & \ddots & & & & \\ & & 2g_{L(n_L)} & & & \\ & & & -2g_{C(1)} & & \\ & & & & \ddots & \\ & & & & & -2g_{C(n_C)} \end{bmatrix} \tag{5.11}$$

The branch voltages ($\underline{v}_b(t)$) can be related to the node voltages ($\underline{v}(t)$) as in (5.12).

$$\underline{v}_b(t) = \mathbf{A_i}\, \underline{v}(t) \tag{5.12}$$

Here, $\mathbf{A_i}$ is the node incidence matrix [60] which relates the node voltages to the branch voltages of the energy storage elements. At time t the relation between the node voltages $\underline{v}(t)$ and the branch history currents $\underline{h}_b(t)$ is given by (5.13).

$$\mathbf{Y}\, \underline{v}(t) = -\mathbf{A_i}^T\, \underline{h}_b(t) \tag{5.13}$$

Here, $\mathbf{Y}$ is the nodal admittance matrix. Combining (5.10) - (5.13), we get (5.14).

$$\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \underline{h}_b(t+\Delta t) \\ \underline{v}(t+\Delta t) \end{bmatrix} = \begin{bmatrix} \mathbf{K} & \mathbf{Y_b A_i} \\ \mathbf{A_i}^T & \mathbf{Y} \end{bmatrix} \begin{bmatrix} \underline{h}_b(t) \\ \underline{v}(t) \end{bmatrix} \tag{5.14}$$

Here, $\mathbf{I}$ is an identity matrix of size $(n_L + n_C) \times (n_L + n_C)$. Eq. (5.14) is of the form:

$$\mathbf{M}\, \underline{X}(t+\Delta t) = \mathbf{H}\, \underline{X}(t) \tag{5.15}$$

Eq. (5.15) is the required discrete-time state-space representation of a circuit in the descriptor form. An illustrative example is provided in Appendix F to verify the above procedure.

5.5.2 Multi-rate EMT Simulations using the Multi Area Thevenin Equivalent (MATE) Concept [2]

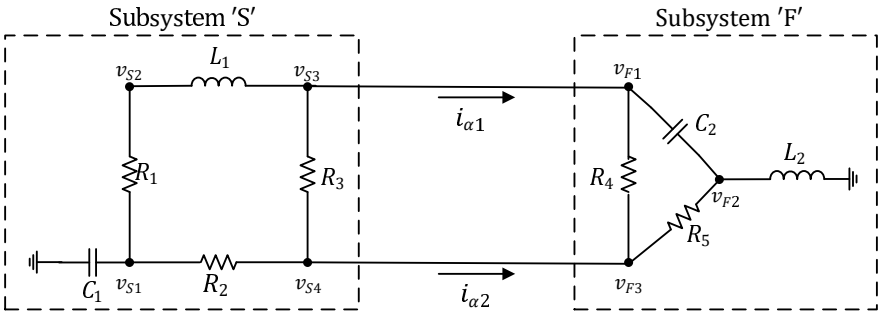
In this section, we review the details of the Multi Area Thevenin Equivalent (MATE)-based multi-rate simulation method [2]. This method is used later for two purposes: (a) to explain how to set up the equations for an approach for stability assessment of multi-rate EMT simulations; (b) to demonstrate the effectiveness of the proposed approach in accurately predicting the stability of multi-rate EMT simulations.

Firstly, we review how MATE helps in partitioning a network in single-rate EMT sim-

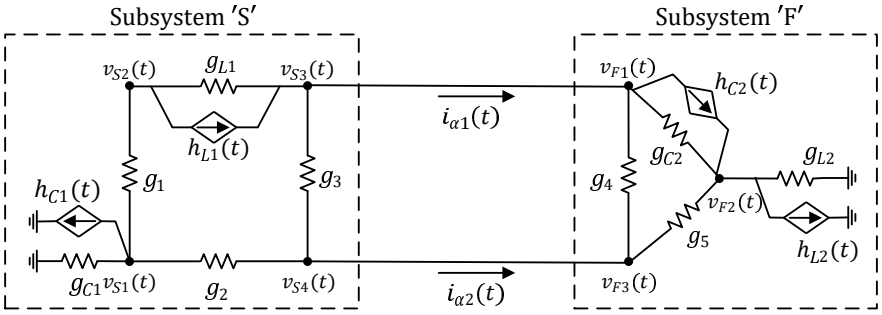
ulations [30]. Later on, we review how this approach is extended by *Moreira et al.* [2] for carrying out multi-rate EMT simulations.

Single-rate EMT Simulation using MATE [30]

MATE is a network partitioning and recombination technique in which the solution of the entire system is decomposed into the solutions of smaller subsystems and the solution of a few links that interconnect these subsystems [2]. To understand how MATE can be used for single-rate EMT simulations, let us consider the circuit shown in Fig. 5.8.



(a) A Sample Network



(b) Companion Circuit for the Sample Network of Fig. 5.8a

Figure 5.8: Sample Network for explaining the MATE formulation for single-rate EMT simulations

The circuit consists of two subsystems 'S' and 'F' interconnected by two links. Without loss of generality, it is assumed that the links have zero impedance. Let:

$\mathbf{Y_S}$, $\mathbf{Y_F}$: Admittance matrices of subsystems 'S' and 'F'

$\underline{v}_S$, $\underline{v}_F$: Node voltage vectors for subsystems 'S' and 'F'

$\underline{h}_{bS}$, $\underline{h}_{bF}$: Branch history current vectors for subsystems 'S' and 'F'

$i_{\alpha1}$, $i_{\alpha2}$: Link currents

$$\underline{v}_S = [v_{S1} \quad v_{S2} \quad v_{S3} \quad v_{S4}]^T; \quad \underline{h}_{bS} = [h_{L1} \quad h_{C1}]^T$$
$$\underline{v}_F = [v_{F1} \quad v_{F2} \quad v_{F3}]^T; \quad \underline{h}_{bF} = [h_{L2} \quad h_{C2}]^T$$
$$\underline{i}_\alpha = [i_{\alpha1} \quad i_{\alpha2}]^T$$

The branch equations for the links are:

$$\text{Link 1: } v_{S3}(t) - v_{F1}(t) = 0$$
$$\text{Link 2: } v_{S4}(t) - v_{F3}(t) = 0$$

Compactly, these can be written as in (5.16).

$$\mathbf{P_S}^T \underline{v}_S(t) + \mathbf{P_F}^T \underline{v}_F(t) = 0 \tag{5.16}$$

Here, $\mathbf{P_S}$ and $\mathbf{P_F}$ are incidence matrices [60] that relate the internal node voltages of each subsystem to the branch voltages of the links. The nodal equations for subsystems 'S' and 'F' are as in (5.17) and (5.18) respectively.

$$\mathbf{Y_S}\, \underline{v}_S(t) + \mathbf{P_S}\, \underline{i}_\alpha(t) = -\mathbf{A_{iS}}^T \underline{h}_{bS}(t) \tag{5.17}$$
$$\mathbf{Y_F}\, \underline{v}_F(t) + \mathbf{P_F}\, \underline{i}_\alpha(t) = -\mathbf{A_{iF}}^T \underline{h}_{bF}(t) \tag{5.18}$$

Here, $\mathbf{A_{iS}}$ and $\mathbf{A_{iF}}$ are the node incidence matrices [60] which relate the internal node voltages to the branch voltages of the energy storage elements. Combining (5.16) - (5.18) yields (5.19).

$$\left[\begin{array}{c|c|c} \mathbf{Y_S} & \mathbf{0} & \mathbf{P_S} \\ \hline \mathbf{0} & \mathbf{Y_F} & \mathbf{P_F} \\ \hline \mathbf{P_S}^T & \mathbf{P_F}^T & \mathbf{0} \end{array}\right] \left[\begin{array}{c} \underline{v}_S(t) \\ \hline \underline{v}_F(t) \\ \hline \underline{i}_\alpha(t) \end{array}\right] = \left[\begin{array}{c} -\mathbf{A_{iS}}^T \underline{h}_{bS}(t) \\ \hline -\mathbf{A_{iF}}^T \underline{h}_{bF}(t) \\ \hline 0 \end{array}\right] \tag{5.19}$$

By performing a few elementary matrix manipulations, (5.19) is transformed to (5.20).

$$\left[\begin{array}{c:c:c} \mathbf{I_S} & \mathbf{0} & \mathbf{B_S} \\ \hdashline \mathbf{0} & \mathbf{I_F} & \mathbf{B_F} \\ \hdashline \mathbf{0} & \mathbf{0} & \mathbf{Z_{th}} \end{array}\right] \left[\begin{array}{c} \underline{v}_S(t) \\ \hdashline \underline{v}_F(t) \\ \hdashline \underline{i}_\alpha(t) \end{array}\right] = \left[\begin{array}{c} \underline{e}_S(t) \\ \hdashline \underline{e}_F(t) \\ \hdashline \underline{V}_{thS}(t) + \underline{V}_{thF}(t) \end{array}\right] \tag{5.20}$$

Eq. (5.20) essentially translates to the following:

$$\underline{i}_\alpha(t) = \mathbf{Z_{th}}^{-1} \left(\underline{V}_{thS}(t) + \underline{V}_{thF}(t)\right) \tag{5.21}$$

$$\underline{v}_S(t) = \underline{e}_S(t) - \mathbf{B_S}\, \underline{i}_\alpha(t) \tag{5.22}$$

$$\underline{v}_F(t) = \underline{e}_F(t) - \mathbf{B_F}\, \underline{i}_\alpha(t) \tag{5.23}$$

Here,

$$\underline{e}_S(t) = -\mathbf{Y_S}^{-1}\mathbf{A_{iS}}^T\, \underline{h}_{bS}(t) \tag{5.24}$$

$$\underline{e}_F(t) = -\mathbf{Y_F}^{-1}\mathbf{A_{iF}}^T\, \underline{h}_{bF}(t) \tag{5.25}$$

$$\underline{V}_{thS}(t) = \mathbf{P_S}^T\, \underline{e}_S(t) \tag{5.26}$$

$$\underline{V}_{thF}(t) = \mathbf{P_F}^T\, \underline{e}_F(t) \tag{5.27}$$

$$\mathbf{B_S} = \mathbf{Y_S}^{-1}\mathbf{P_S} \tag{5.28}$$

$$\mathbf{B_F} = \mathbf{Y_F}^{-1}\mathbf{P_F} \tag{5.29}$$

$$\mathbf{Z_{th}} = \mathbf{Z_{thS}} + \mathbf{Z_{thF}} \tag{5.30}$$

where, $\mathbf{Z_{thS}} = \mathbf{P_S}^T\mathbf{B_S}$ and $\mathbf{Z_{thF}} = \mathbf{P_F}^T\mathbf{B_F}$.

$\underline{V}_{thS}$ represents the equivalent Thevenin source of subsystem 'S' whereas $\mathbf{Z_{thS}}$ represents the equivalent Thevenin resistance of subsystem 'S'. Similarly, $\underline{V}_{thF}$ is the equivalent Thevenin source of subsystem 'F' whereas $\mathbf{Z_{thF}}$ represents the equivalent Thevenin resistance of subsystem 'F'. Eq. (5.21) is the generalization for a multi-link case of the system shown

in Fig. 5.9.

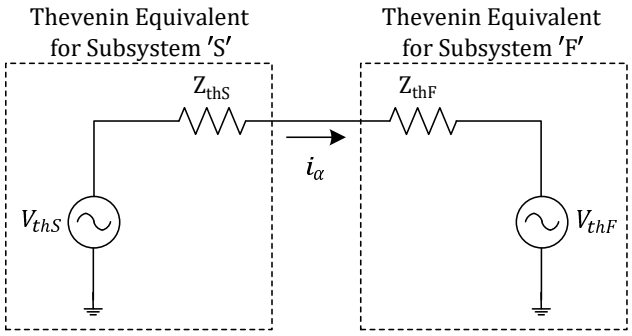


Figure 5.9: Equivalent for (5.21) in the Single-link Case

Using (5.21) - (5.30), the following steps are used for computing the network solution at any time t: [30]

1. $\underline{e}_S(t)$ and $\underline{e}_F(t)$ are computed using (5.24) and (5.25). These are the "partial" solutions (as (5.24) and (5.25) are similar to computing the node voltages of each subsystem assuming link currents are 0).

2. Using the partial solutions, the Thevenin source vectors ($\underline{V}_{thS}(t)$ and $\underline{V}_{thF}(t)$) for each subsystem are computed using (5.26) and (5.27).

3. Knowing $\underline{V}_{thS}(t)$ and $\underline{V}_{thF}(t)$, and $\mathbf{Z_{th}}$ from (5.30), the link currents $\underline{i}_\alpha(t)$ are then computed using (5.21).

4. Knowing the link currents $\underline{i}_\alpha(t)$ and the partial solutions $\underline{e}_S(t)$ and $\underline{e}_F(t)$, the final network solutions $\underline{v}_S(t)$ and $\underline{v}_F(t)$ for subsystems 'S' and 'F' respectively, are computed using (5.22) and (5.23) respectively.

Multi-rate EMT Simulation using MATE [2]

Reference [2] proposes a method that uses the MATE concept for interfacing two subsystems in EMT simulations that are discretized with distinct time-steps. To understand how it works, let us consider the same sample circuit as in Fig. 5.8; however, the subsystem 'S'

(indicating the 'slow' subsystem) is discretized with the large time-step ΔT while the subsystem 'F' (or 'fast' subsystem) is discretized with the small time-step Δt. It is assumed that $\Delta T = N\Delta t$, where N is a positive integer greater than 1.

Because of the distinct time-steps that are used for the two subsystems, [2] proposes to use the following to account for the contributions from each subsystem.

1. *Linear Interpolation of $\underline{V}_{thS}$ at every Δt instant*

   The Thevenin source vector of the slow subsystem (i.e., $\underline{V}_{thS}$) is only updated at instants which are an integer multiple of ΔT. To calculate its values at intermediate Δt instants when only solving the fast subsystem, linear interpolation is used.

2. *Averaging of $\underline{i}_{\alpha}$ over one full ΔT interval*

   The link currents (i.e., $\underline{i}_{\alpha}$) are calculated at each Δt instant within a ΔT interval when only the fast subsystem is solved. To calculate the corresponding current injected into the slow subsystem at the end of a ΔT interval, $\underline{i}_{\alpha}$ is averaged over a ΔT interval.

Details are given below.

Linear Interpolation of $\underline{V}_{thS}$:

The fast subsystem is solved at every intermediate Δt instant with a ΔT interval. The slow subsystem remains latent at these instants. Hence, it is necessary to appropriately account for the contribution of the slow subsystem while solving the fast subsystem. The procedure for doing this, as proposed in [2], is discussed here.

As seen in (5.21), for computing the link currents $\underline{i}_{\alpha}$, only the Thevenin sources $\underline{V}_{thS}$ and $\underline{V}_{thF}$ as well as the Thevenin impedance $[\mathbf{Z_{th}}]$ are needed. Once $\underline{i}_{\alpha}$ are known, the final network solution for the fast subsystem can be computed using (5.23).

From (5.26) and (5.27), we can see that the Thevenin source vectors $\underline{V}_{thS}$ and $\underline{V}_{thF}$ only depend on history values. Therefore, referring to the time-line in Fig. 5.10, if the network solution for the slow subsystem is known at any time t, then the Thevenin source vector $\underline{V}_{thS}(t + \Delta T)$ is automatically known. The Thevenin source vector $\underline{V}_{thS}(t)$ is already known (by construction). Hence, [2] proposes to estimate the values of $\underline{V}_{thS}$ at the every

intermediate Δt instants using linear interpolation using the two known values $\underline{V}_{thS}(t)$ and $\underline{V}_{thS}(t+\Delta T)$ as illustrated in Fig. 5.10.

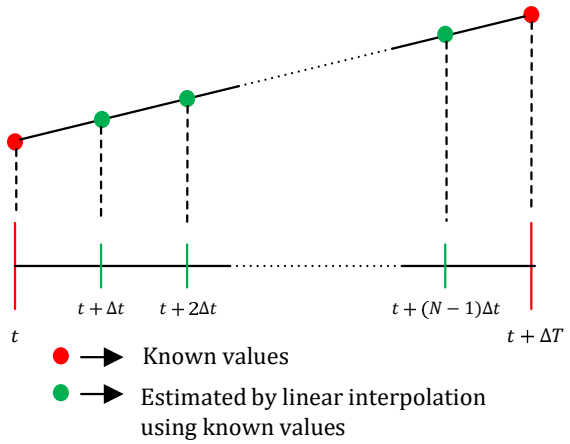


Figure 5.10: Estimating V_{thS} at intermediate steps by linear interpolation

In summary, the procedure at any intermediate Δt instant within a ΔT interval is:

1. Estimate $\underline{V}_{thS}$ by linear interpolation using the two known values at the two large time-step instants.
2. Compute $\underline{V}_{thF}$ using (5.27).
3. Compute $\underline{i}_{\alpha}$ using (5.21).
4. Compute $\underline{v}_{F}$ using (5.23).

Averaging of $\underline{i}_{\alpha}$:

At every time instant which is an integer multiple of ΔT, the entire system's solution is computed. This requires that both the fast as well as the slow subsystems be solved. The fast subsystem has been updated N times within the ΔT interval (i.e., at every intermediate Δt instant). If only the results at the $N\Delta t$ instant were transferred to the slow subsystem, all the intermediate solutions of the fast subsystem would be ignored. Hence, [2] proposes to use the following steps:

1. Knowing $\underline{V}_{thS}(t+\Delta T)$ and $\underline{V}_{thF}(t+\Delta T)$, $\underline{i}_{\alpha}(t+\Delta T)$ is computed using (5.21).

2. The $\underline{v}_F(t+\Delta T)$ is computed using (5.23) using $\underline{i}_\alpha(t+\Delta T)$ from Step 1.

3. For computing $\underline{v}_S(t+\Delta T)$,

   - The average of $\underline{i}_\alpha(t+\Delta T)$ and $\underline{i}_\alpha$'s over the previous $(N-1)$ intermediate Δt instants is computed.
   - This average value is substituted in (5.22) to compute $\underline{v}_S(t+\Delta T)$.

5.5.3 Time-invariant Representation for MATE-based Multi-rate EMT Simulation Method

As discussed at the start of Section 5.5, for obtaining a time-invariant representation for any arbitrary multi-rate simulation method, the first step is to write down the governing equations for each small time-step (Δt) instant within the large time-step (ΔT) interval. For the MATE-based multi-rate simulation method [2], these equations can be written down based on the discussion in Section 5.5.2.

Consider a general time interval $(t,\ t+\Delta T)$ as shown in Fig. 5.11.

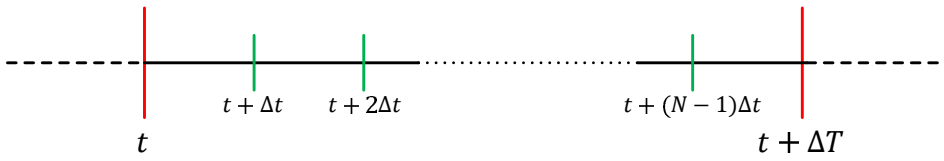


Figure 5.11: Timeline for Obtaining a Time-invariant Representation for MATE-based Multi-rate EMT Simulation Method

At any large time-step instant t:

At this instant (indicated by red in Fig. 5.11), both the slow subsystem as well as the fast subsystem are solved. The equations used to solve both these subsystems at this instant can be obtained using (5.21) - (5.30). This yields (5.31) - (5.35).

$$\underline{V}_{thS}(t) = -\mathbf{P_S}^T \mathbf{Y_S}^{-1} \mathbf{A_{iS}}^T \underline{h}_{bS}(t) \tag{5.31}$$

$$\underline{V}_{thF}(t) = -\mathbf{P_F}^T \mathbf{Y_F}^{-1} \mathbf{A_{iF}}^T \, \underline{h}_{bF}(t) \tag{5.32}$$

$$\underline{i}_{\alpha}(t) = \mathbf{Z_{th}}^{-1} \left(\underline{V}_{thS}(t) + \underline{V}_{thF}(t) \right) \tag{5.33}$$

$$\underline{v}_F(t) = -\mathbf{Y_F}^{-1} \mathbf{A_{iF}}^T \, \underline{h}_{bF}(t) - \mathbf{Y_F}^{-1} \mathbf{P_F} \, \underline{i}_{\alpha}(t) \tag{5.34}$$

$$\underline{v}_S(t) = -\mathbf{Y_S}^{-1} \mathbf{A_{iS}}^T \, \underline{h}_{bS}(t) - \mathbf{Y_S}^{-1} \mathbf{P_S} \left\{ \frac{1}{N} \sum_{r=1}^{N} \underline{i}_{\alpha}(t - \Delta T + r\Delta t) \right\} \tag{5.35}$$

On the right-hand side of (5.35), $\underline{i}_{\alpha}(t)$ is known from (5.33) whereas $\underline{i}_{\alpha}(t - \Delta T + r\Delta t)$ for $r = 1, 2, \ldots N-1$ are the known values of $\underline{i}_{\alpha}$ at the intermediate Δt instants in the previous ΔT interval. In (5.33), the $\mathbf{Z_{th}}$ is as in (5.36).

$$\begin{aligned} \mathbf{Z_{th}} &= \quad \mathbf{Z_{thS}} \quad + \quad \mathbf{Z_{thF}} \\ &= \mathbf{P_S}^T \mathbf{Y_S}^{-1} \mathbf{P_S} \; + \; \mathbf{P_F}^T \mathbf{Y_F}^{-1} \mathbf{P_F} \end{aligned} \tag{5.36}$$

Note: For calculating $\mathbf{Z_{thS}}$ of the slow subsystem in (5.36), the admittance matrix $\mathbf{Y_S}$ uses ΔT as the time-step. Whereas for calculating $\mathbf{Z_{thF}}$ of the fast subsystem in (5.36), the admittance matrix $\mathbf{Y_F}$ uses Δt as the time-step.

Now, let $\underline{h}_{bS}$ be the vector of branch history current for the storage elements (i.e., inductors, capacitors) in the slow subsystem; and let $\underline{h}_{bF}$ be the vector of branch history currents for the storage elements in the fast subsystem. Now, knowing $\underline{v}_S(t)$ and $\underline{v}_F(t)$ from (5.35) and (5.34) respectively, the branch history current vectors $\underline{h}_{bS}$ and $\underline{h}_{bF}$ can be updated as in (5.37) - (5.38).

$$\underline{h}_{bS}(t + \Delta T) = \mathbf{K_S} \, \underline{h}_{bS}(t) + \mathbf{Y_{bS}} \mathbf{A_{iS}} \, \underline{v}_S(t) \tag{5.37}$$

$$\underline{h}_{bF}(t + \Delta t) = \mathbf{K_F} \, \underline{h}_{bF}(t) + \mathbf{Y_{bF}} \mathbf{A_{iF}} \, \underline{v}_F(t) \tag{5.38}$$

These equations are similar to the branch history current updating equation in (5.10). However, notice that $\underline{h}_{bS}$ is updated at $(t + \Delta T)$ i.e., the next large time-step instant, whereas $\underline{h}_{bF}$ is updated at $(t + \Delta t)$ i.e., the next small time-step instant. This is because the slow subsystem is solved directly at the next large time-step instant whereas the fast subsystem is solved at the next small time-step instant.

Using a few algebraic manipulations, it can be shown that (5.31) - (5.35) are equivalent to (5.39) - (5.42). See Appendix G for the proof.

$$\mathbf{Y_S}\,\underline{v}'_S(t) + \mathbf{P_S}\,\underline{i}_\alpha(t) + \mathbf{A_{iS}}^T\,\underline{h}_{bS}(t) = \underline{0} \tag{5.39}$$

$$\mathbf{Y_F}\,\underline{v}_F(t) + \mathbf{P_F}\,\underline{i}_\alpha(t) + \mathbf{A_{iF}}^T\,\underline{h}_{bF}(t) = \underline{0} \tag{5.40}$$

$$\mathbf{P_S}^T\underline{v}'_S(t) + \mathbf{P_F}^T\underline{v}_F(t) = \underline{0} \tag{5.41}$$

$$\mathbf{Y_S}\,\underline{v}_S(t) + \mathbf{P_S}\left\{\frac{1}{N}\sum_{r=1}^{N}\underline{i}_\alpha(t-\Delta T+r\Delta t)\right\} + \mathbf{A_{iS}}^T\,\underline{h}_{bS}(t) = \underline{0} \tag{5.42}$$

Here, $\underline{v}'_S(t)$ is a dummy variable that arises from the algebraic manipulations whereas $\underline{v}_S(t)$ is the node voltage vector for the slow subsystem. Together, (5.37) - (5.42) are the governing equations at any time instant which is an integer multiple of ΔT.

Now, knowing $\underline{h}_{bS}(t+\Delta T)$ from (5.37), $\underline{V}_{thS}(t+\Delta T)$ can be computed using (5.43).

$$\underline{V}_{thS}(t+\Delta T) = -\mathbf{P_S}^T\mathbf{Y_S}^{-1}\mathbf{A_{iS}}^T\,\underline{h}_{bS}(t+\Delta T) \tag{5.43}$$

This $\underline{V}_{thS}(t+\Delta T)$ is required while estimating the value of $\underline{V}_{thS}$ by linear interpolation at the subsequent Δt instants (discussed below).

At every intermediate small time-step instant $(t+k\Delta t)$, where $k = 1, 2, \ldots N-1$:

At these instants (indicated by green in Fig. 5.11), only the fast subsystem is solved. The contribution of the slow subsystem is accounted for by interpolating the value of $\underline{V}_{thS}$ using two of its known values at t and $(t+\Delta T)$ [2]. Thus, the equations used for solving the fast subsystem at each of these instants are as given in (5.44) - (5.47).

$$\underline{V}_{thS}(t+k\Delta t) = \underline{V}_{thS}(t) + \frac{k}{N}\left(\underline{V}_{thS}(t+\Delta T) - \underline{V}_{thS}(t)\right) \tag{5.44}$$

$$\underline{V}_{thF}(t+k\Delta t) = -\mathbf{P_F}^T\mathbf{Y_F}^{-1}\mathbf{A_{iF}}^T\,\underline{h}_{bF}(t+k\Delta t) \tag{5.45}$$

$$\underline{i}_\alpha(t+k\Delta t) = \mathbf{Z_{th}}^{-1}\left(\underline{V}_{thS}(t+k\Delta t) + \underline{V}_{thF}(t+k\Delta t)\right) \tag{5.46}$$

$$\underline{v}_F(t+k\Delta t) = -\mathbf{Y_F}^{-1}\mathbf{A_{iF}}^T\,\underline{h}_{bF}(t+k\Delta t) - \mathbf{Y_F}^{-1}\mathbf{P_F}\,\underline{i}_\alpha(t+k\Delta t) \tag{5.47}$$

Note: The $\mathbf{Z_{th}}$ used in (5.46) is the same as in (5.36). It does not change over the course of the entire simulation as the simulated circuit is LTI in the continuous time-domain.

Now, knowing $\underline{v}_F(t+k\Delta t)$ from (5.47), the branch history current vector $\underline{h}_{bF}(t+(k+1)\Delta t)$ at the next Δt instant can be updated as in (5.48).

$$\underline{h}_{bF}(t+(k+1)\Delta t) = \mathbf{K_F}\ \underline{h}_{bF}(t+k\Delta t) + \mathbf{Y_{bF}A_{iF}}\ \underline{v}_F(t+k\Delta t) \tag{5.48}$$

Again using a few algebraic manipulations, it can be shown that (5.44) - (5.47) are equivalent to (5.49) - (5.52). See Appendix G for the proof.

$$\underline{h}_{bS}(t+k\Delta t) = \underline{h}_{bS}(t) + \frac{k}{N}\left(\underline{h}_{bS}(t+\Delta T) - \underline{h}_{bS}(t)\right) \tag{5.49}$$

$$\mathbf{Y_S}\ \underline{v}'_S(t+k\Delta t) + \mathbf{P_S}\ \underline{i}_\alpha(t+k\Delta t) + \mathbf{A_{iS}}^T\ \underline{h}_{bS}(t+k\Delta t) = \underline{0} \tag{5.50}$$

$$\mathbf{Y_F}\ \underline{v}_F(t+k\Delta t) + \mathbf{P_F}\ \underline{i}_\alpha(t+k\Delta t) + \mathbf{A_{iF}}^T\ \underline{h}_{bF}(t+k\Delta t) = \underline{0} \tag{5.51}$$

$$\mathbf{P_S}^T\underline{v}'_S(t+k\Delta t) + \mathbf{P_F}^T\underline{v}_F(t+k\Delta t) = \underline{0} \tag{5.52}$$

Here, $\underline{v}'_S(t+k\Delta t)$ are dummy variables that arise from the algebraic manipulations. Together, (5.48) - (5.52) are the governing equations at each intermediate Δt instant.

Based on the above discussion, we now know all the governing equations at each Δt instant within a ΔT interval. For obtaining a time-invariant representation for the MATE-based multi-rate simulation method [2], we define $\underline{X}(t)$ as in (5.53).

$$\underline{X}(t) := \begin{bmatrix} \underline{X}_S(t) \\ \underline{v}_S(t) \\ \underline{X}_F(t) \\ \underline{i}_\alpha(t) \\ \underline{X}_L(t) \\ \underline{X}_L(t-\Delta T) \end{bmatrix} \tag{5.53}$$

In (5.53), $\underline{v}_S(t)$ is the node voltage vector for the slow subsystem while $\underline{i}_\alpha(t)$ is the vector

of link currents. $\underline{X}_S(t)$, $\underline{X}_F(t)$ and $\underline{X}_L(t)$ are as defined in (5.54). Note that $\underline{X}_S(t)$, $\underline{X}_F(t)$ and $\underline{X}_L(t)$ have a form which is similar to the lifted state vector given in (5.5).

$$\underline{X}_S(t) := \begin{bmatrix} \underline{h}_{bS}(t) \\ \underline{v}'_S(t) \\ \underline{h}_{bS}(t+\Delta t) \\ \underline{v}'_S(t+\Delta t) \\ \vdots \\ \underline{h}_{bS}(t+(N-1)\Delta t) \\ \underline{v}'_S(t+(N-1)\Delta t) \end{bmatrix}; \quad \underline{X}_F(t) := \begin{bmatrix} \underline{h}_{bF}(t) \\ \underline{v}_F(t) \\ \underline{h}_{bF}(t+\Delta t) \\ \underline{v}_F(t+\Delta t) \\ \vdots \\ \underline{h}_{bF}(t+(N-1)\Delta t) \\ \underline{v}_F(t+(N-1)\Delta t) \end{bmatrix}; \quad \underline{X}_L(t) := \begin{bmatrix} \underline{i}_\alpha(t+\Delta t) \\ \underline{i}_\alpha(t+2\Delta t) \\ \vdots \\ \underline{i}_\alpha(t+(N-1)\Delta t) \end{bmatrix} \tag{5.54}$$

Now, using (5.37) - (5.42) as well as (5.48) - (5.52), and the $\underline{X}(t)$ defined in (5.53), we can get the required time-invariant representation for a MATE-based multi-rate simulation of an LTI circuit as in (5.55).

$$\mathbf{M}\,\underline{X}(t+\Delta T) = \mathbf{H}\,\underline{X}(t) \tag{5.55}$$

$$\Longrightarrow \left[\begin{array}{c:c:c:c:c:c} \mathbf{M_{11}} & & & & & \\ \hdashline & & & & & \\ \hdashline & & \mathbf{M_{33}} & & & \\ \hdashline & & & & & \\ \hdashline & & & & & \\ \hdashline & & & & & \mathbf{I} \end{array}\right] \left[\begin{array}{c} \underline{X}_S(t+\Delta T) \\ \hdashline \underline{v}_S(t+\Delta T) \\ \hdashline \underline{X}_F(t+\Delta T) \\ \hdashline \underline{i}_\alpha(t+\Delta T) \\ \hdashline \underline{X}_L(t+\Delta T) \\ \hdashline \underline{X}_L(t) \end{array}\right]$$

$$= \left[\begin{array}{c:c:c:c:c:c} \mathbf{H_{11}} & \mathbf{H_{12}} & & \mathbf{H_{14}} & \mathbf{H_{15}} & \\ \hdashline & \mathbf{H_{22}} & & \mathbf{H_{24}} & & \mathbf{H_{26}} \\ \hdashline & & \mathbf{H_{33}} & \mathbf{H_{34}} & \mathbf{H_{35}} & \\ \hdashline \mathbf{H_{41}} & & \mathbf{H_{43}} & & & \\ \hdashline \mathbf{H_{51}} & & \mathbf{H_{53}} & & & \\ \hdashline & & & & \mathbf{I} & \end{array}\right] \left[\begin{array}{c} \underline{X}_S(t) \\ \hdashline \underline{v}_S(t) \\ \hdashline \underline{X}_F(t) \\ \hdashline \underline{i}_\alpha(t) \\ \hdashline \underline{X}_L(t) \\ \hdashline \underline{X}_L(t-\Delta T) \end{array}\right] \tag{5.56}$$

In (5.56), all entries which are blank are zero while **I** is an identity matrix of appropriate size. The entries in other sub-matrices of **M** and **H** can be populated using (5.37) - (5.42) and (5.48) - (5.52). The table below presents which equations can be used for populating the entries in these sub-matrices.

Sub-matrices of M and H	**Equation to be Used**
$\mathbf{M_{11}}$, $\mathbf{H_{11}}$, $\mathbf{H_{12}}$, $\mathbf{H_{14}}$, $\mathbf{H_{15}}$	(5.37), (5.39), (5.49) and (5.50)
$\mathbf{H_{22}}$, $\mathbf{H_{24}}$, $\mathbf{H_{26}}$	(5.42)
$\mathbf{M_{33}}$, $\mathbf{H_{33}}$, $\mathbf{H_{34}}$, $\mathbf{H_{35}}$	(5.38), (5.40), (5.48) and (5.51)
$\mathbf{H_{41}}$, $\mathbf{H_{43}}$	(5.41)
$\mathbf{H_{51}}$, $\mathbf{H_{53}}$	(5.52)

Using the time-invariant model of (5.55), we can say that a MATE-based multi-rate simulation of an LTI circuit will be stable if all the finite eigenvalues of the matrix pencil $(\mathbf{H}, \mathbf{M})$ are inside the unit circle. However, computing all the eigenvalues can be onerous (especially for large systems). Hence, in this work we assess the stability of the simulation by only computing the finite eigenvalue with the largest magnitude and looking at whether it is inside or outside the unit circle. The `eigs()` function in MATLAB [105] is used for directly computing the finite eigenvalue of $(\mathbf{H}, \mathbf{M})$ which has the largest magnitude.

5.6 Validation Tests

To demonstrate that the proposed method can accurately predict the stability of multi-rate EMT simulations of LTI circuits, we consider two test cases in this section. Both test cases use the MATE-based multi-rate EMT simulation method proposed in [2]. No particular commercial simulation package was used for the simulations; instead, stand-alone programs were written in MATLAB [105].

5.6.1 Test Case 1: Effect of Time-steps on Stability of Simulation

The first test case is same as the example presented in Section 5.3. For convenience, the circuit is repeated here in Fig. 5.12.

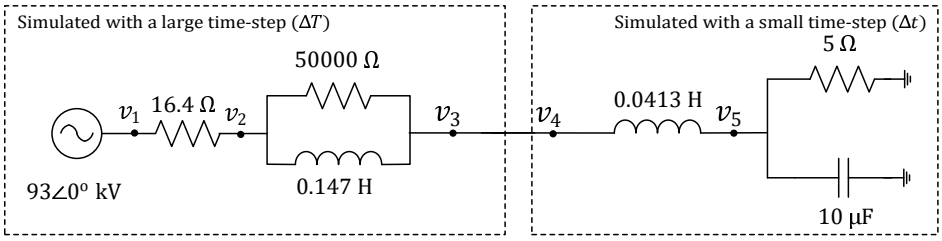


Figure 5.12: Circuit for Test Case 1 (Same as in Fig. 5.2)

Three scenarios are tested:

1. With $\Delta T = 50\ \mu s$ and $\Delta t = 5\ \mu s$.
2. With $\Delta T = 50\ \mu s$ and $\Delta t = 10\ \mu s$.
3. With $\Delta t = 5\ \mu s$ while ΔT is varied from Δt to $10\Delta t$ in increments of Δt.

A time-invariant model for the simulation is obtained using the procedure discussed in Section 5.5.

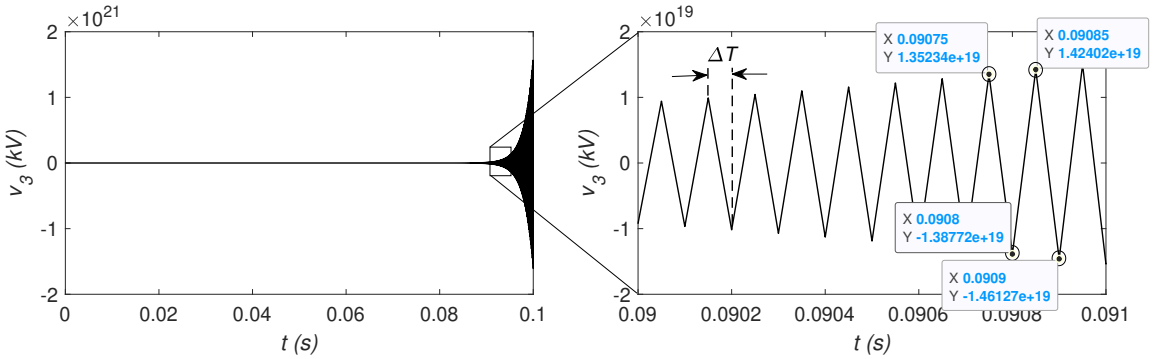
Scenario 1 - With $\Delta T = 50\ \mu s$ and $\Delta t = 5\ \mu s$:

This is the same scenario that is presented in Section 5.3. The results in Fig. 5.4 showed that the simulation with $\Delta T = 50\ \mu s$ and $\Delta t = 5\ \mu s$ is unstable. In this section, we explore whether this can be analytically verified using the proposed approach.

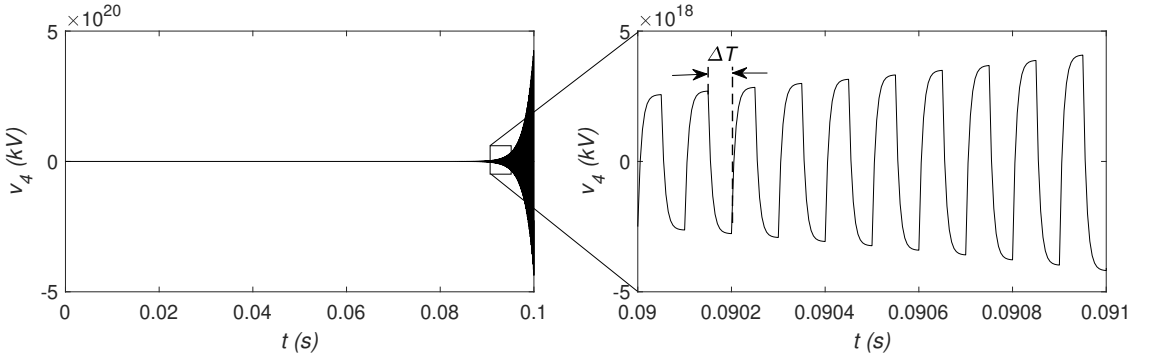
Using the time-invariant model for the simulated system, the finite eigenvalue of $(\mathbf{H}, \mathbf{M})$ with the largest magnitude is found to be -1.02615. We can see that this eigenvalue is outside the unit circle, thus showing that the proposed approach also predicts that the simulation will be unstable.

To corroborate that the result obtained using the proposed approach is indeed correct, we can look at the waveforms for the voltages at nodes 3 and 4 given in Fig. 5.13. In the magnified plot of v_3 given in Fig. 5.13a, we can see that the ratio of the next sample to the

previous sample at any ΔT instant is −1.02615. This is precisely equal to the largest magnitude finite eigenvalue that is computed using the proposed stability assessment approach.



(a) Node Voltage v_3 (from ΔT side)



(b) Node Voltage v_4 (from Δt side)

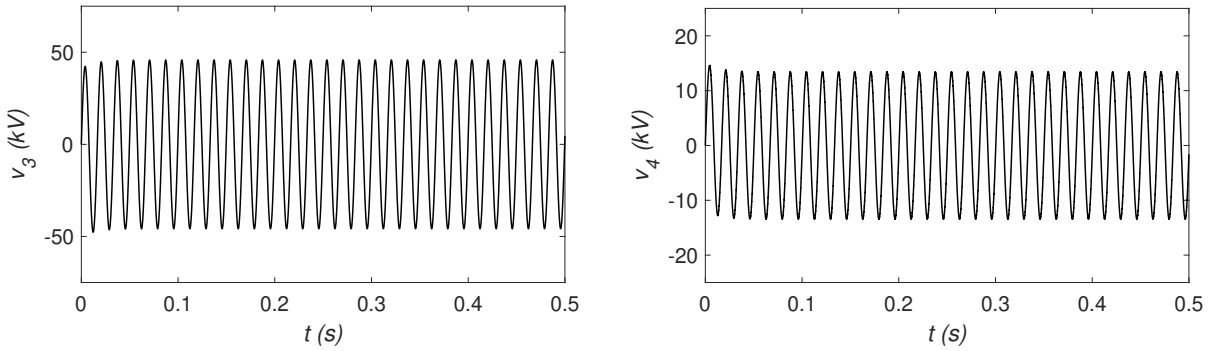
Figure 5.13: Test Case 1 - Voltages at Node 3 & 4 (with ΔT = 50 μs and Δt = 5 μs)

Additionally, from the magnified plots of both v_3 and v_4, we can also verify the periodicity property (with period = ΔT) exhibited by multi-rate EMT simulations in the discrete time-domain as was discussed in Section 5.4.

Scenario 2 - With ΔT = 50 μs and Δt = 10 μs:

In the second scenario, the same circuit is simulated with ΔT = 50 μs and Δt = 10 μs. In this case, the finite eigenvalue of $(\mathbf{H}, \mathbf{M})$ with the largest magnitude is 0.99636 which is inside the unit circle thus implying that the simulation is stable.

The waveforms for the voltages at node 3 (from ΔT side) and node 4 (from Δt side) are shown in Fig. 5.14. These are indeed stable and thus corroborate this fact.



(a) Node Voltage v_3 (from ΔT side) **(b) Node Voltage v_4 (from Δt side)**

Figure 5.14: Test Case 1 - Voltages at Node 3 & 4 (with ΔT = 50 μs and Δt = 10 μs)

Scenario 3 - With Δt = 5 μs while ΔT is varied from Δt to $10\Delta t$:

In the first two scenarios, both ΔT and Δt were kept constant. Now, in contrast, Δt is kept constant at 5 μs whereas ΔT is varied from 5 μs (i.e., = Δt) to 50 μs (i.e., = $10\Delta t$) in increments of 5 μs. For this, the locus of all the finite eigenvalues of $(\mathbf{H}, \mathbf{M})$ is as shown in Fig. 5.15 using the time-invariant model for the simulation.

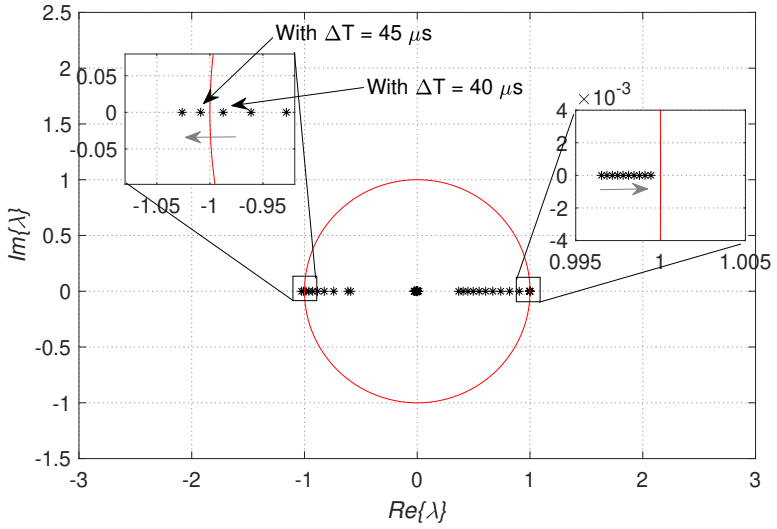
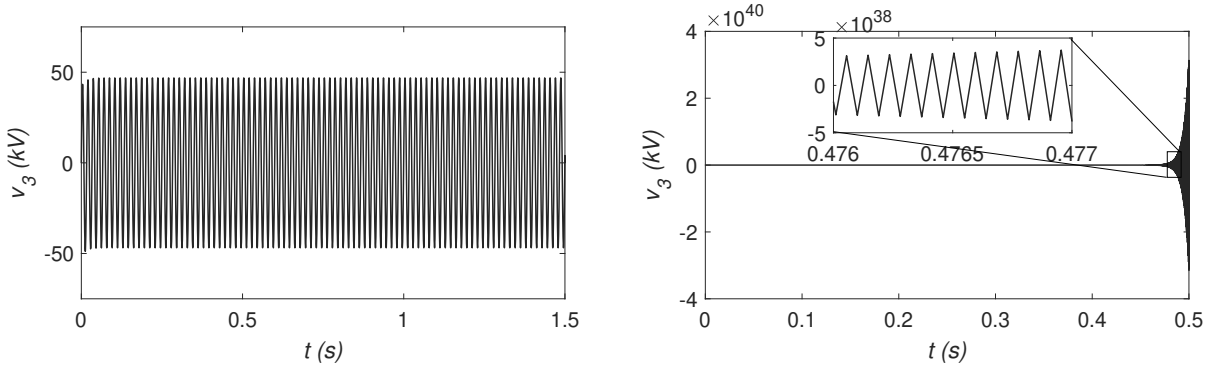


Figure 5.15: Test Case 1, Scenario 3 - Locus of the Finite Eigenvalues of $(\mathbf{H}, \mathbf{M})$ when ΔT is varied

From Fig. 5.15, we can see that for $\Delta T \leq 40$ μs, all the eigenvalues are inside the unit circle which means the simulation will be stable. Whereas, for $\Delta T \geq 45$ μs, one of the

eigenvalues goes outside the unit circle implying that the simulation will be unstable.

To corroborate this result, we run two separate simulations; one with $\Delta T = 40\ \mu s$ and the other with $\Delta T = 45\ \mu s$ while keeping Δt constant at $5\ \mu s$ in both cases. The waveforms of the node voltage v_3 for both these scenarios is shown in Fig. 5.16. It is clear that with $\Delta T = 40\ \mu s$, the simulation is stable whereas with $\Delta T = 45\ \mu s$, it is unstable. This precisely agrees with the results obtained using the proposed approach.



(a) With $\Delta T = 40\ \mu s$ and $\Delta t = 5\ \mu s$ **(b) With $\Delta T = 45\ \mu s$ and $\Delta t = 5\ \mu s$**

Figure 5.16: Test Case 1, Scenario 3 - Voltage at Node 3

5.6.2 Test Case 2: Effect of Circuit Parameters on Stability of Simulation

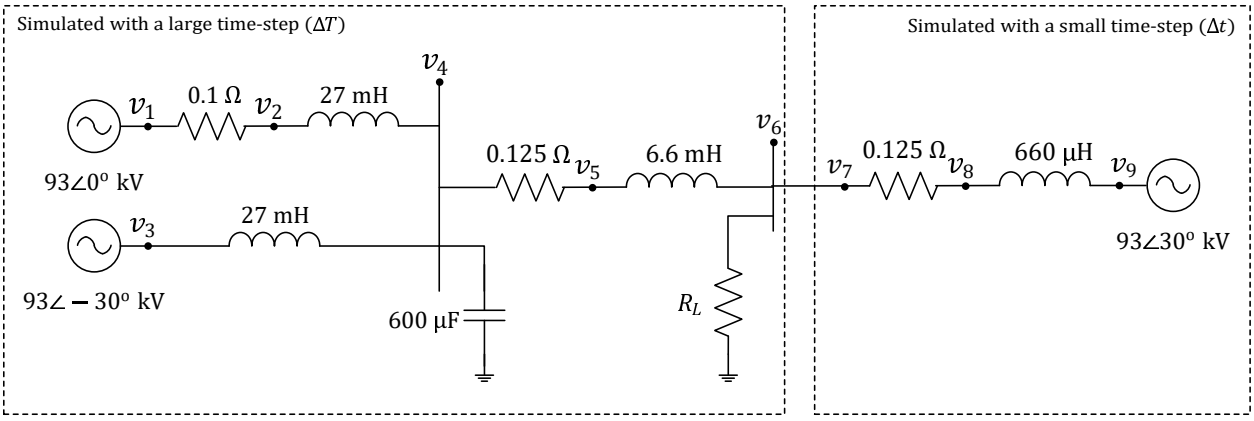


Figure 5.17: Circuit for Test Case 2

In the first test case, we showed how time-steps can affect the stability of the simula-

tion and also how the proposed approach is effective in accurately predicting the stability properties of the simulation in such cases. In this test case, we aim to investigate whether the proposed approach can predict the stability of a multi-rate simulation when a circuit parameter is varied across multiple simulation runs. For this, consider the circuit shown in Fig. 5.17.

A large time-step ΔT = 50 μs is used for the left side sub-network while a small time-step Δt = 1 μs is used for the right side sub-network. A time-invariant model for the simulation is obtained using the procedure discussed in Section 5.5. The value of the resistance R_L is varied from 2 $k\Omega$ to 4 $k\Omega$ in steps of 0.1 $k\Omega$. For this, the locus of all the finite eigenvalues of $(\mathbf{H}, \mathbf{M})$ is as shown in Fig. 5.18. The plot shows that for $R_L \leq 3.3\ k\Omega$, all the eigenvalues are inside the unit circle which means the simulation will be stable; whereas, for $R_L \geq 3.4\ k\Omega$, one of the eigenvalues goes outside the unit circle implying that the simulation will be unstable.

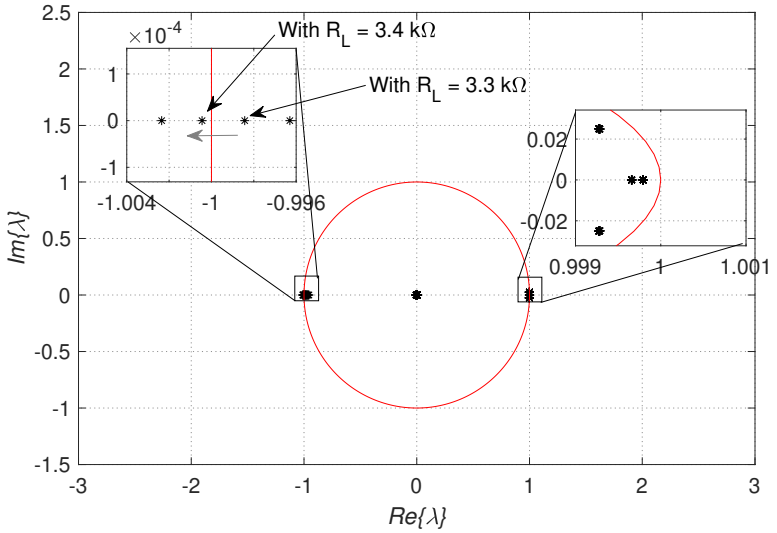
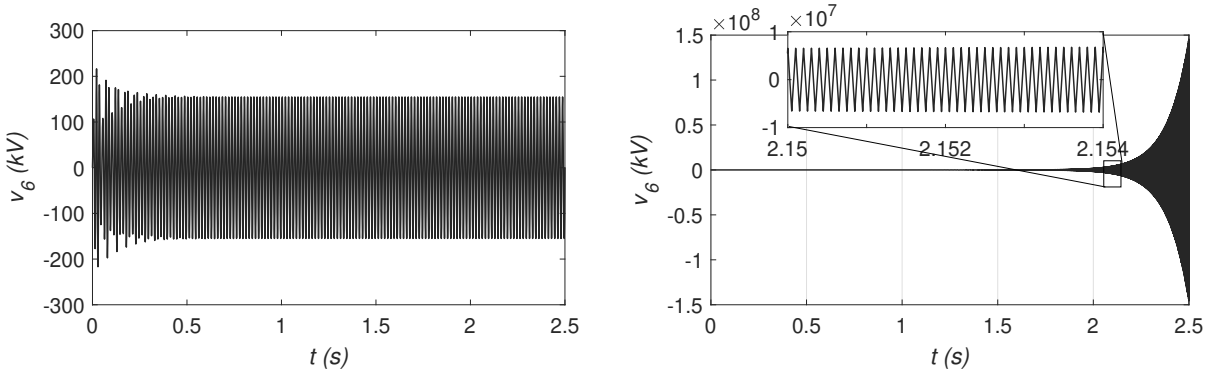


Figure 5.18: Test Case 2 - Locus of the Finite Eigenvalues of $(\mathbf{H}, \mathbf{M})$ when R_L is varied

To corroborate this result, we run two separate simulation scenarios; one with R_L = 3.3 $k\Omega$ and the other with R_L = 3.4 $k\Omega$. The waveforms of the node voltage v_6 for both these scenarios is shown in Fig. 5.19. It is clear that with R_L = 3.3 $k\Omega$, the simulation is stable

whereas with R_L = 3.4 $k\Omega$, it is unstable. This precisely agrees with the results obtained using the proposed approach.



(a) With R_L = 3.3 Ω **(b) With R_L = 3.4 Ω**

Figure 5.19: Test Case 2 - Voltage at Node 6 (with ΔT = 50 μs and Δt = 1 μs)

Thus, the above two test cases verify that the proposed stability assessment approach is able to accurately predict the stability of multi-rate EMT simulations of LTI circuits.

5.7 Conclusions

This chapter developed a novel approach for assessing the stability of multi-rate EMT simulations of linear time-invariant (LTI) circuits. Firstly, it is demonstrated that multi-rate EMT simulations can produce unstable results for stable continuous-time LTI circuits even when the trapezoidal integration method is used for discretizing various components in each subsystem. Further, it is shown that such simulations always yield a periodically varying system in the discrete-time domain. By exploiting this property and using the 'lifting' technique, a sampled data time-invariant representation, in the form $\mathbf{M}\underline{X}(t + \Delta T) = \mathbf{H}\underline{X}(t)$, is obtained for the simulated discrete-time system. The numerical stability of the simulation can then be assessed by looking at the finite eigenvalues of the matrix pencil $(\mathbf{H}, \mathbf{M})$. The test cases that are presented successfully demonstrate the effectiveness of the proposed method in accurately predicting the stability of multi-rate EMT simulations of LTI circuits.

5.8 Contributions

1. Demonstrated that multi-rate EMT simulations can produce unstable results for stable continuous-time LTI circuits even when the trapezoidal integration method is used for discretizing various components in each subsystem.

2. Showed that multi-rate simulations of LTI circuits always yield a periodically varying system in the discrete-time domain.

3. Obtained a sampled data time-invariant representation for the simulated discrete-time system By exploiting this property using the well-known technique of 'lifting' [56] and used it for assessing the numerical stability of the simulation using eigenvalue analysis.

4. Effectively validated the proposed method to study the effect of varying the time-step ratio ($\Delta T/\Delta t$), and a circuit parameter on the stability of multi-rate EMT simulations.

Chapter 6

Contributions, Conclusions and Future Work

This chapter discusses the major conclusions of this thesis as well as provides some recommendations for future work.

6.1 Contributions and Conclusions

In this thesis, novel methods were developed that help in decomposing the simulated network in order to accelerate EMT simulations using parallel processing as well as the stability and computational efficiency of some of the existing methods in the literature have been analyzed.

- In Chapter 2, an alternative method was investigated for formulating the state variable equations of a circuit for EMT simulations using Descriptor State-space Equations (DSE).
  - A step-by-step procedure was presented for automatically formulating the DSEs using a circuit's netlist. The formulation is easier to implement as a computer algorithm compared to the classical state-space equations.
  - For carrying out EMT simulation studies, the formulated DSEs were discretized using the trapezoidal integration method.

  - The DSE-based approach was compared with the widely used Companion Circuits (CC) approach. One of the advantages of using the DSE-based formulation is that in addition to running EMT simulations, it is possible to analytically calculate the eigenvalues of the network directly. However, it has the disadvantage of having a marginally higher run-time than the CC-based approach for EMT simulations.
  - A procedure for interfacing a DSE-based formulation with a CC-based EMT simulator was also presented. This enables the interfacing of arbitrary power networks with a commercial CC-based EMT simulation package without the need for building it in that package. Moreover, this combined approach also allows easy parallel simulation as multiple DSE-based modules can be run on separate processors and then interfaced with a CC-based EMT simulator.

- In Chapter 3, two of the commonly used sparse matrix-based parallelization methods were adapted and used for accelerating DSE-based EMT simulations. Both methods relied on transforming the sparse matrix $\mathbf{M} = \left(\mathbf{E} + \frac{\mathbf{A}\Delta t}{2}\right)$ to a form which is amenable to parallel processing.
  - In Method 1, $\mathbf{M}$ was automatically transformed to a block-diagonal form. This method can only be used when the simulated system contains distributed parameter lines and/or cables.
  - In Method 2, $\mathbf{M}$ was automatically transformed $\mathbf{M}$ to a bordered block diagonal form. This method is more general in its application as it only uses the sparsity pattern of the matrix for its transformation (i.e., it does not require the presence of any special elements like distributed parameter lines/cables in the simulated system).
  - For comparing the computational performance of the two methods, two test cases based on the footprints of real-world systems were used. The first case was a 500-bus test system while the second one was a 2000-bus test system. Comparative studies showed that Method 2 was able to provide higher simulation speedup.

- In Chapter 4, a novel delay-based interface was developed which permits partitioning of the system at branches where series inductors are present. As series inductors are widespread in a power network, this greatly increases the number of partitioning opportunities.

  - The work reported in this chapter used time-domain simulations backed up by the frequency response plots of the admittance matrix elements of the interface to show that it has better accuracy compared to the widely used stub-line interface.
  - As the proposed method always generates a strictly passive interface, it does not contribute negatively to the stability of the simulation.
  - The resulting sub-network partitions are interfaced with each other in a straightforward manner using voltage sources that depend only on history values and permit speed-up of the simulation by parallelizing the network solution.
  - Simulation test cases with HVdc converters were used to demonstrate the practical utility of this interface. It is also demonstrated that the proposed approach facilitates implementation on a parallel computing platform without compromising the numerical stability of the simulation.

- In Chapter 5, a novel approach was developed for assessing the stability of multi-rate EMT simulations of linear time-invariant (LTI) circuits.

  - It should be recalled that single-rate simulations using the trapezoidal integration method are always stable when the simulated circuit is a stable continuous-time LTI system. In contrast, in this chapter, it was clearly shown that multi-rate EMT simulations of a stable continuous-time LTI circuit may become unstable even when the trapezoidal integration method is used in each subsystem.
  - It was shown that multi-rate simulation of LTI circuits always yields a periodically varying in the discrete-time domain. By exploiting this property and using the 'lifting' technique, a sampled data time-invariant representation, in the form

$\mathbf{M}\underline{X}(t + \Delta T) = \mathbf{H}\underline{X}(t)$, is obtained for the simulated discrete-time system. Following this, the numerical stability of the simulation can be assessed by looking at the finite eigenvalues of the matrix pencil $(\mathbf{H}, \mathbf{M})$.

- The test cases that were presented successfully demonstrated the effectiveness of the proposed method in accurately predicting the stability of multi-rate EMT simulations of LTI circuits.

The following publication resulted from the work reported in this thesis:

Ajinkya Sinkar, Huanfeng Zhao, Bolin Qu, and Aniruddha M. Gole, "A Comparative Study of Electromagnetic Transient Simulations using Companion Circuits, and Descriptor State-space Equations," *Electric Power System Research*, vol. 198, Sept. 2021, 107360.

6.2 Recommendations for Future Work

Some of the possible avenues for extending the work reported in this thesis are given below:

1. **With regards to "DSE-based EMT simulations"**:

   - Switches can be included in the DSE-based formulation of Chapter 2 so that the approach can be tested with networks containing power electronic converters. However, this would also require adapting and implementing the following techniques in a DSE-based EMT simulator:
     - Interpolation, reinitialization and chatter removal techniques [74–76].
     - Methods for efficiently refactorizing a sparse matrix every time a switch changes in conduction state [69, 70].
   - The scalability of the matrix-level parallelization approaches used in Chapter 3 for accelerating DSE-based EMT simulations can be tested by using a computing platform that has a higher number of processor cores than the one that was used in this thesis.

- In Chapter 2, it was shown that DSE-based EMT simulations have a slightly higher run-time compared to CC-based EMT simulations when simulating the same network. It will be worthwhile to compare the performance of a matrix-level parallelized DSE-based EMT simulator with that of a corresponding CC-based EMT simulator. That way, one can find out if the performance gap between the two approaches gets bridged when parallel computing techniques are used.

2. **With regards to "UPNP interface":**

   - In Chapter 4, although it is shown that the UPNP interface is adequately accurate, other passivity guaranteeing topologies *may* be possible. It is worthwhile to explore if there is an alternative to the proposed UPNP interface that is both guaranteed passive and has a better accuracy performance.

3. **With regards to "Stability Assessment of Multi-rate EMT Simulations":**

   - Using the developed method in Chapter 5 as a starting point, the development of a stability assessment method for multi-rate EMT simulation of LTI circuits where the ratio $\Delta T/\Delta t$ is NOT an integer can be explored.
   - The development of a method for assessing the stability of multi-rate EMT simulations of networks with switches is an interesting extension that is worth exploring.

Appendix A

Details of the LCC-HVdc link used in Chapter 2 and Chapter 4

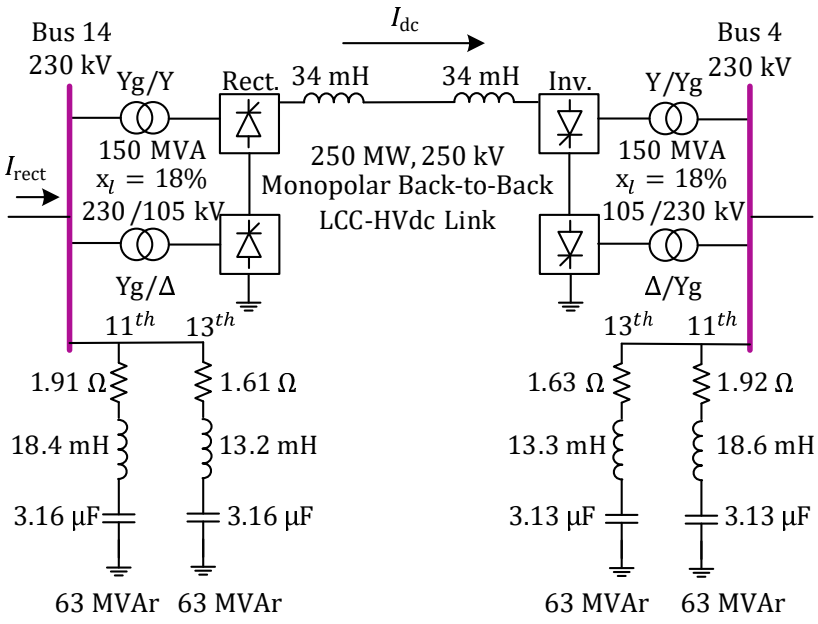


Figure A.1: Back-to-back LCC-HVdc Link

1. The rectifier side uses constant dc link current control (set-point = 1.0 pu).

2. The inverter side uses constant extinction control (set-point = 18^o for a 60 Hz system).

Appendix B

Including Distributed Parameter Transmission Lines in a DSE-based Formulation

For DSE-based EMT simulations, the equations of the circuit are formulated in the continuous time-domain using MNA in the form given in (B.1).

$$\mathbf{E}\underline{\dot{x}} = -\mathbf{A}\underline{x} + \mathbf{B}\underline{u} \tag{B.1}$$

Here,

$$\underline{x} = \begin{bmatrix} \underline{v}_N^T & \underline{i}_L^T & \underline{i}_S^T \end{bmatrix}^T \qquad \underline{u} = \begin{bmatrix} \underline{v}_S^T & \underline{j}_S^T \end{bmatrix}^T$$

$\underline{v}_N$: Node voltages.

$\underline{i}_L$: Inductor currents.

$\underline{i}_S$: Currents through independent voltage sources.

$\underline{v}_S$: Independent voltage sources.

$\underline{j}_S$: Independent current sources.

Expanding (B.1) gives (B.2).

$$\begin{bmatrix} \mathbf{C} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{L} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \frac{d}{dt} \begin{bmatrix} \underline{v}_N \\ \underline{i}_L \\ \underline{i}_S \end{bmatrix} = - \begin{bmatrix} \mathbf{G} & \mathbf{A_L} & \mathbf{A_{vs}} \\ \mathbf{A_L^T} & \mathbf{0} & \mathbf{0} \\ \mathbf{A_{vs}^T} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \underline{v}_N \\ \underline{i}_L \\ \underline{i}_S \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{A_{js}} \\ \mathbf{0} & \mathbf{0} \\ -\mathbf{I} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \underline{v}_S \\ \underline{j}_S \end{bmatrix} \tag{B.2}$$

Here,

C: Capacitance matrix.

L: Diagonal matrix containing inductance values.

G: Conductance matrix (corresponding to lumped resistors in the circuit)

$\mathbf{A_L}$: Incidence matrix for inductor branches.

$\mathbf{A_{vs}}$: Incidence matrix for voltage source branches.

$\mathbf{A_{js}}$: Incidence matrix for current source branches.

The circuit for the Bergeron model of a single-phase transmission line is as shown in Fig. B.1.

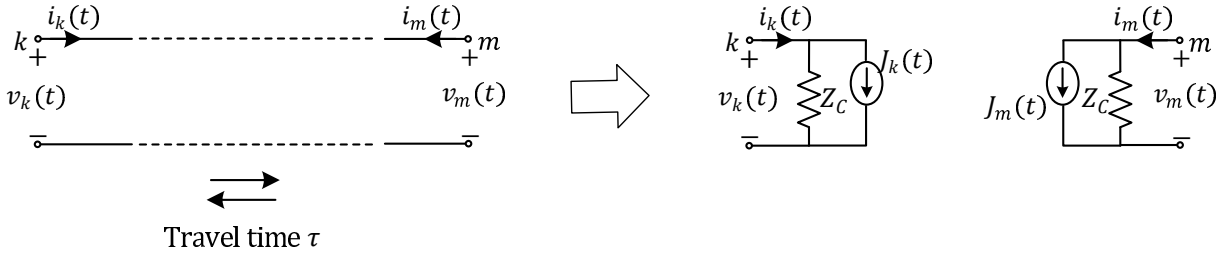


Figure B.1: Bergeron Model of a Single-phase Transmission Line

In this model, the values of the current sources $J_k(t)$ and $J_m(t)$ at time t are given by [84]:

$$\begin{aligned} J_k(t) &= -i_m(t-\tau) - \frac{1}{Z_C} v_m(t-\tau) \\ J_m(t) &= -i_k(t-\tau) - \frac{1}{Z_C} v_k(t-\tau) \end{aligned} \tag{B.3}$$

As we can see in (B.3), $J_k(t)$ and $J_m(t)$ at time t depend on the values of the terminal voltages v_k and v_m, and the port currents i_k and i_m at time $(t-\tau)$, where τ is the transport delay of the transmission line.

Thus, a single-phase transmission line modelled using the Bergeron model can be easily included in the DSE-based formulation by doing the following[1]:

1. Add the Z_C shown in Fig. B.1 to the sub-matrix $\mathbf{G}$ in (B.2).

2. Make the current sources $J_k(t)$ and $J_m(t)$ shown in Fig. B.1 a part of the $\underline{j}_S$ vector in (B.2).

And since all the node voltages are part of the $\underline{x}$ vector (as shown in (B.2)), the current sources $J_k(t)$ and $J_m(t)$ can be easily calculated at each time-step by reading $v_k(t-\tau)$ and $v_m(t-\tau)$ for each line from the $\underline{x}(t-\tau)$ vector.

NOTE: $i_k(t-\tau)$ and $i_m(t-\tau)$, which are required for calculating $J_k(t)$ and $J_m(t)$, can be easily computed once we know $v_k(t-\tau)$ and $v_m(t-\tau)$ as well as $J_k(t-\tau)$ and $J_m(t-\tau)$. This is done by applying KCL at the two ends of the line model shown in Fig. B.1.

[1]Note, this approach can be easily extended to any other distributed parameter model of a transmission line or cable (single-phase or three-phase) by using its corresponding equations.

Appendix C

Example to Illustrate the BBD Conversion Procedure from Section 3.4.2

For illustrating the BBD conversion procedure shown in Fig. 3.6, consider the circuit shown in Fig. C.1.

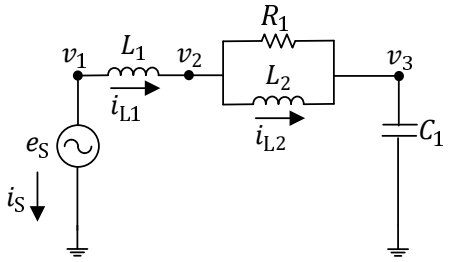


Figure C.1: A Simple Circuit to Illustrate BBD Conversion using METIS Library

The Descriptor State-space Equations (DSEs) for this circuit can be written in the form:

$$\mathbf{E}\underline{\dot{x}} = -\mathbf{A}\underline{x} + \mathbf{B}\underline{u} \tag{C.1}$$

where,

$\underline{x} = [v_1 \quad v_2 \quad v_3 \quad i_{L1} \quad i_{L2} \quad i_S]^T$; $\quad \underline{u} = [e_S]$

The matrices $\mathbf{E}$, $\mathbf{A}$ and $\mathbf{B}$ are as follows:

$$\mathbf{E} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & C_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & L_1 & 0 & 0 \\ 0 & 0 & 0 & 0 & L_2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}; \quad \mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & -1 \\ 0 & \frac{1}{R_1} & -\frac{1}{R_1} & -1 & 1 & 0 \\ 0 & -\frac{1}{R_1} & \frac{1}{R_1} & 0 & -1 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}; \quad \mathbf{B} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -1 \end{bmatrix}$$

Based on these, the matrix $\mathbf{M} = \left(\mathbf{E} + \dfrac{\mathbf{A}\Delta t}{2}\right)$ will have the following non-zero pattern:

$$\mathbf{M} = \begin{bmatrix} & & & m_1 & & m_2 \\ & m_3 & m_4 & m_5 & m_6 & \\ & m_7 & m_8 & & m_9 & \\ m_{10} & m_{11} & & m_{12} & & \\ & m_{13} & m_{14} & & m_{15} & \\ m_{16} & & & & & \end{bmatrix}$$

STEP 1: Permute Rows of M to have Zero-free Main Diagonal

Since there is an ideal voltage source between node 1 and ground in the circuit of Fig. C.1, there are zeros on the main diagonal of $\mathbf{M}$ at position (1,1) as well as position (6,6). In order to have a zero-free main diagonal, we exchange rows 1 and 6 of $\mathbf{M}$ to give $\mathbf{M}_1$ as shown:

$$\mathbf{M}_1 = \begin{bmatrix} m_{16} & & & & & \\ & m_3 & m_4 & m_5 & m_6 & \\ & m_7 & m_8 & & m_9 & \\ m_{10} & m_{11} & & m_{12} & & \\ & m_{13} & m_{14} & & m_{15} & \\ & & & m_1 & & m_2 \end{bmatrix}$$

STEP 2: Construct the Graph of $\mathbf{M}_1 + \mathbf{M}_1^T$

The graph corresponding to the non-zero pattern of $\mathbf{M}_1 + \mathbf{M}_1^T$ is as shown in Fig. C.2.

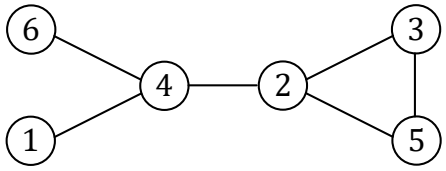


Figure C.2: Graph of $\mathbf{M}_1 + \mathbf{M}_1^T$

STEP 3: Partition the Graph using METIS Library [83]

Suppose we want two internal blocks (i.e., $n = 2$). If METIS [44] is used to partition the graph shown in Fig. C.2 into two parts, then the resulting partitioned graph is as shown in Fig. C.3.

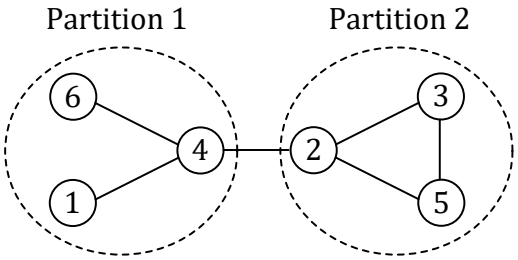


Figure C.3: Partitioned Graph

STEP 4: Find the Row Permutation Matrix using the Partitioned Graph

To find the permutation matrix using the partitioned graph of Fig. C.3, we use the following steps:

1. For each partition k (where $k = 1, 2, \cdots n$, n being the number of partitions), form a set $\mathbb{S}_k$ containing its "internal nodes", and another set $\mathbb{E}$ containing its "edge nodes".

    - *Internal nodes*: Nodes in a partition for which all its neighbouring nodes are in its own partition.

      e.g: In Fig. C.3, nodes 1 and 6 are internal nodes of partition 1 where as nodes 3 and 5 are internal nodes of partition 2.

- *Edge nodes*: Nodes in a partition for which at least one of its neighbouring node is not from its own partition.
  e.g: In Fig. C.3, node 4 is an edge node in Partition 1 whereas node 2 is an edge node in Partition 2.

For the graph in Fig. C.3, $n = 2$. Therefore there will be a total of four sets viz. $\mathbb{S}_1$, $\mathbb{E}_1$ and $\mathbb{S}_2$, $\mathbb{E}_2$. Each of these sets for the graph in Fig. C.3 is as given below:

$$\begin{aligned} \mathbb{S}_1 &= \{1, 6\} \\ \mathbb{E}_1 &= \{4\} \\ \mathbb{S}_2 &= \{3, 5\} \\ \mathbb{E}_2 &= \{2\} \end{aligned} \tag{C.2}$$

2. Arrange the elements in each set $\mathbb{E}_k$ in the descending order of their "external degree".

   - *External degree of an edge node* = Number of neighbours of a node that are not in its own partition.
     e.g: In Fig. C.3, external degree of node 4 is equal to 1 (since only one of its neighbours, node 2 is not in its own partition). The same can be said about node 2.

   For the graph in Fig. C.3, there are only one element in each of $\mathbb{E}_1$ and $\mathbb{E}_2$ (given in (C.2)). Therefore, they are already in the descending order of their external degrees.

3. Build a new boundary node set $\mathbb{B}$ by doing the following:

   (a) Initialize $k = 1$.

   (b) If $\mathbb{E}_k$ is empty, directly go to (d). If it is not empty, then set x equal to the first node in $\mathbb{E}_k$, remove this first node from $\mathbb{E}_k$ and put it in $\mathbb{B}$.

   (c) Remove every neighbour j of node x from each $\mathbb{E}_i$ (where $i = 1, 2, \cdots n$) and return j to its corresponding internal node set $\mathbb{S}_i$.

(d) Increment k by 1. If $k > n$, reset $k = 1$. Go back to (b).

The above process is stopped when $\mathbb{E}$ for every partition is empty.

By following this procedure for the graph in Fig. C.3, we will get:

$$\begin{aligned} \mathbb{S}_1 &= \{1, 6\} \\ \mathbb{S}_2 &= \{3, 5, 2\} \\ \mathbb{B} &= \{4\} \end{aligned} \tag{C.3}$$

The corresponding graph is given in Fig. C.4.

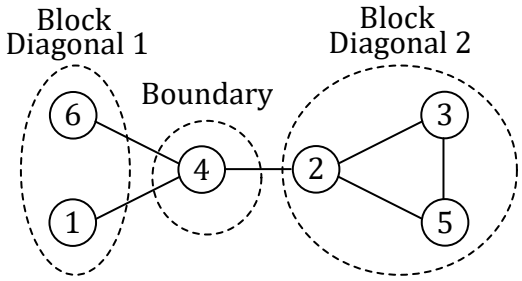


Figure C.4: Partitioned Graph with Internal and Boundary Nodes Marked

4. Build the set $\mathbb{A} = \left(\bigcup_{i=1}^{n} \mathbb{S}_i\right) \cup \mathbb{B}$. The permutation vector (p) is then:

$$\underline{p} = [p_1 \;\; p_2 \;\; \cdots \;\; p_N]$$

where $p_i = i^{th}$ element in $\mathbb{A}$ (where $i = 1, 2, \cdots n_n$, n_n being the number of nodes in the graph). And using this permutation vector, the corresponding permutation matrix $\mathbf{P}_{bbd}$ can be built using the following[1]:

$$\mathbf{P}_{bbd} = \text{sparse}(1{:}N,\ \underline{p},\ 1) \tag{C.4}$$

For the graph in Fig. C.3,

$$\underline{p} = [1,\ 6,\ 3,\ 5,\ 2,\ 4]$$

[1]Note, (C.4) is given in MATLAB-like notation.

Hence,

$$\mathbf{P}_{bbd} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \tag{C.5}$$

The matrix $\mathbf{M}_1$ can then be transformed to a bordered block diagonal matrix $\mathbf{M}_{bbd}$ using (C.6).

$$\mathbf{M}_{bbd} = \mathbf{P}_{bbd}\mathbf{M}_1\mathbf{P}_{bbd}^T \tag{C.6}$$

This gives $\mathbf{M}_{bbd}$ in the required BBD form as given below:

$$\mathbf{M}_{bbd} = \left[\begin{array}{cc|ccc|c} M_{16} & & & & & \\ & M_2 & & & & M_1 \\ \hline & & M_8 & M_9 & M_7 & \\ & & M_{14} & M_{15} & M_{13} & \\ & & M_4 & M_6 & M_3 & M_5 \\ \hline M_{10} & & & & M_{11} & M_{12} \end{array}\right]$$

An important thing to note in (C.6) is that the permutation $\mathbf{P}_{bbd}$ is applied to $\mathbf{M}_1$, not $\mathbf{M}$.

Appendix D

Equivalence of Circuits in Fig. 4.7a and Fig. 4.8 in Chapter 4

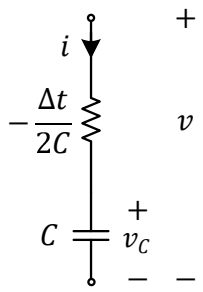


Figure D.1: R-C branch in the Fig. 4.8

In Fig. D.1, let v_c be the voltage across the capacitor, v be the voltage across the entire R-C branch and i be the current through the capacitor. Then by KCL and KVL, we know that:

$$C\frac{dv_c}{dt} = i \tag{D.1}$$

$$v = \left(-\frac{\Delta t}{2C}\right) i + v_c \tag{D.2}$$

By discretizing (D.1) using trapezoidal method, we get:

$$v_c(t) = \left(\frac{\Delta t}{2C}\right) i(t) + \left[\left(\frac{\Delta t}{2C}\right) i(t-\Delta t) + v_c(t-\Delta t)\right] \tag{D.3}$$

Substituting $v_c(t)$ and $v_c(t-\Delta t)$ from (D.2) into (D.3) gives the value of $V_h(t)$ given below.

$$\begin{aligned} V_h(t) &= v(t) \\ &= v(t-\Delta t) + \left(\frac{\Delta t}{C}\right) i(t-\Delta t) \end{aligned} \tag{D.4}$$

This is exactly the same expression that we would if a capacitor C is discretized using the Forward Euler method. Hence, we can conclude that the circuit shown in Fig. 4.8 is equivalent to the one in Fig. 4.7a.

Appendix E

Derivation of (4.10)

Let $\mathbf{M} \equiv \mathbf{Z}(j\omega) + \mathbf{Z}^H(j\omega)$. Then, the matrix $\mathbf{M}$ for the UPNP interface of Fig. 4.9 is of the form:

$$\mathbf{M} = \begin{bmatrix} M_{11} & M_{12} \\ M_{12} & M_{11} \end{bmatrix} \tag{E.1}$$

Here, using (4.8) and (4.9), we get:

$$M_{11} = 4\Delta t \left[\frac{\omega^2 L}{(\Delta t)^2 \omega^2 + 16} \right] + \frac{2L}{\Delta t} \left[\frac{2 - (\Delta t)^2 \omega^2}{(\Delta t)^2 \omega^2 + 16} \right] \tag{E.2}$$

$$M_{12} = \frac{2L}{\Delta t} \left[\frac{2 - (\Delta t)^2 \omega^2}{(\Delta t)^2 \omega^2 + 16} \right] \tag{E.3}$$

Let

$$a \equiv 4\Delta t \left[\frac{\omega^2 L}{(\Delta t)^2 \omega^2 + 16} \right]; \quad b \equiv \frac{2L}{\Delta t} \left[\frac{2 - (\Delta t)^2 \omega^2}{(\Delta t)^2 \omega^2 + 16} \right] \tag{E.4}$$

$\therefore$ The matrix $\mathbf{M}$ becomes:

$$\mathbf{M} = \begin{bmatrix} a+b & b \\ b & a+b \end{bmatrix} \tag{E.5}$$

Then, the eigenvalues of $\mathbf{M}$ are:

$$\lambda_1 = a; \quad \lambda_2 = a + 2b \tag{E.6}$$

Substituting the values of a and b from (E.4) into (E.6) gives the required expressions for λ_1 and λ_2 (as given in (4.10)).

Appendix F

Example to Verify the Procedure given in Section 5.5.1

To verify the method discussed in Section 5.5.1 that is used for obtaining a discrete-time state-space representation in the descriptor form directly from the companion circuit formulation of a network, we consider the example circuit shown in Fig. F.1a. Its corresponding companion circuit in discrete time domain is shown in Fig. F.1b. For this example, $L_1 = 1\ \mu H$, $C_1 = 100\ \mu F$, $R_2 = 0.1\ \Omega$, $L_2 = 1\ \mu H$, $C_2 = 1\ \mu F$ and $\Delta t = 2\ \mu s$.

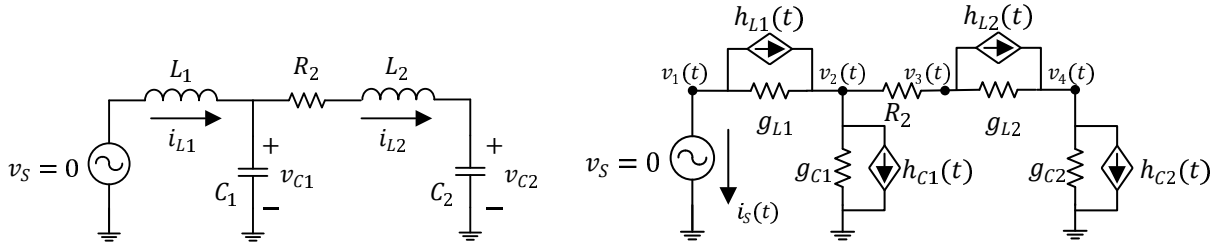


Figure F.1: Verification Example 1

If we write down the standard state space equations for this circuit in the continuous time domain in the form $\dot{\underline{x}}_S = \mathbf{A_S}\underline{x}_S$ with the state vector $\underline{x}_S = [i_{L1} \quad v_{C1} \quad i_{L2} \quad v_{C2}]^T$, then we get

the state matrix $\mathbf{A_S}$ as:

$$\mathbf{A_S} = \begin{bmatrix} 0 & -\dfrac{1}{L_1} & 0 & 0 \\ \dfrac{1}{C_1} & 0 & -\dfrac{1}{C_1} & 0 \\ 0 & \dfrac{1}{L_2} & -\dfrac{R_2}{L_2} & -\dfrac{1}{L_2} \\ 0 & 0 & \dfrac{1}{C_2} & 0 \end{bmatrix}$$

Following this, we can find the corresponding state matrix $\mathbf{G}$ (for the trapezoidal method) in the discrete time-domain using the following expression:

$$\mathbf{G} = \left(\mathbf{I} - \frac{A_S \Delta}{2}\right)^{-1} \left(\mathbf{I} + \frac{A_S \Delta}{2}\right) \tag{F.1}$$

Now, for the circuit of Fig. F.1b, let the branch history current vector $\underline{h}_b(t)$ and the node voltage vector $\underline{v}(t)$ be as follows:

$$\underline{h}_b(t) = \begin{bmatrix} h_{L1}(t) & h_{C1}(t) & h_{L2}(t) & h_{C2}(t) \end{bmatrix}^T$$
$$\underline{v}(t) = \begin{bmatrix} v_1(t) & v_2(t) & v_3(t) & v_4(t) & i_S(t) \end{bmatrix}^T$$

In order to get the discrete-time state-space representation of (5.14) directly from Dommel's formulation for the circuit of Fig. F.1b, we need the matrices $\mathbf{K}$, $\mathbf{Y_b}$, $\mathbf{A_i}$ and $\mathbf{Y}$. These matrices for the circuit of Fig. F.1b are given below:

$$\mathbf{K} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \qquad \mathbf{A_i} = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\mathbf{Y_b} = \begin{bmatrix} 2g_{L1} & 0 & 0 & 0 \\ 0 & -2g_{C1} & 0 & 0 \\ 0 & 0 & 2g_{L2} & 0 \\ 0 & 0 & 0 & -2g_{C2} \end{bmatrix}$$

$$\mathbf{Y} = \begin{bmatrix} g_{L1} & -g_{L1} & 0 & 0 & 1 \\ -g_{L1} & \left(g_{L1} + g_{C1} + \dfrac{1}{R_2}\right) & -\dfrac{1}{R_2} & 0 & 0 \\ 0 & -\dfrac{1}{R_2} & \left(\dfrac{1}{R_2} + g_{L2}\right) & -g_{L2} & 0 \\ 0 & 0 & -g_{L2} & g_{L2} + g_{C2} & 0 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

With these matrices known, the $\mathbf{M}$ and $\mathbf{H}$ matrices can be formulated using the expressions given in (5.14). Table F.1 gives a comparison of the eigenvalues of the matrix $\mathbf{G}$ (from F.1) and the eigenvalues of the matrix pencil $(\mathbf{H}, \mathbf{M})$.

Table F.1: Eigenvalues for the circuit of Fig. F.1

$\sigma(\mathbf{H}, \mathbf{M})$	$\sigma(\mathbf{G})$
$0.9803 \pm 0.1970j$	$0.9803 \pm 0.1970j$
$-0.0048 \pm 0.9514j$	$-0.0048 \pm 0.9514j$
$\infty,\ \infty,\ \infty,\ \infty,\ \infty$	

We can clearly see in Table F.1 that the eigenvalues exactly match (the five extra ∞ eigenvalues in $\sigma(\mathbf{H}, \mathbf{M})$ arise only because $\mathbf{M}$ is rank deficient by 5). Thus, this verifies the procedure discussed in Section 5.5.1.

Appendix G

Proofs Related to the Discussion in Section 5.5.3

G.1 Proof that (5.31) - (5.35) are equivalent to (5.39) - (5.42)

We start with (5.39) - (5.42). Here, these are repeated in (G.1) - (G.4).

$$\mathbf{Y_S}\,\underline{v}'_S(t) + \mathbf{P_S}\,\underline{i}_\alpha(t) + \mathbf{A_{iS}}^T\,\underline{h}_{bS}(t) = \underline{0} \tag{G.1}$$

$$\mathbf{Y_F}\,\underline{v}_F(t) + \mathbf{P_F}\,\underline{i}_\alpha(t) + \mathbf{A_{iF}}^T\,\underline{h}_{bF}(t) = \underline{0} \tag{G.2}$$

$$\mathbf{P_S}^T\underline{v}'_S(t) + \mathbf{P_F}^T\underline{v}_F(t) = \underline{0} \tag{G.3}$$

$$\mathbf{Y_S}\,\underline{v}_S(t) + \mathbf{P_S}\left\{\frac{1}{N}\sum_{r=1}^{N}\underline{i}_\alpha(t - \Delta T + r\Delta t)\right\} + \mathbf{A_{iS}}^T\,\underline{h}_{bS}(t) = \underline{0} \tag{G.4}$$

Arranging (G.1) - (G.3) in a matrix form yields (G.5).

$$\left[\begin{array}{c|c|c} \mathbf{Y_S} & \mathbf{0} & \mathbf{P_S} \\ \hline \mathbf{0} & \mathbf{Y_F} & \mathbf{P_F} \\ \hline \mathbf{P_S}^T & \mathbf{P_F}^T & \mathbf{0} \end{array}\right]\left[\begin{array}{c} \underline{v}'_S(t) \\ \hline \underline{v}_F(t) \\ \hline \underline{i}_\alpha(t) \end{array}\right] = \left[\begin{array}{c} -\mathbf{A_{iS}}^T\underline{h}_{bS}(t) \\ \hline -\mathbf{A_{iF}}^T\underline{h}_{bF}(t) \\ \hline 0 \end{array}\right] \tag{G.5}$$

By performing a few elementary matrix manipulations, (G.5) can be transformed to (G.6).

$$\left[\begin{array}{c|c|c} \mathbf{I_S} & \mathbf{0} & \mathbf{Y_S}^{-1}\mathbf{P_S} \\ \hline \mathbf{0} & \mathbf{I_F} & \mathbf{Y_F}^{-1}\mathbf{P_F} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{Z_{th}} \end{array}\right]\left[\begin{array}{c} \underline{v}'_S(t) \\ \hline \underline{v}_F(t) \\ \hline \underline{i}_\alpha(t) \end{array}\right] = \left[\begin{array}{c} -\mathbf{Y_S}^{-1}\mathbf{A_{iS}}^T\underline{h}_{bS}(t) \\ \hline -\mathbf{Y_F}^{-1}\mathbf{A_{iF}}^T\underline{h}_{bF}(t) \\ \hline \underline{V}_{thS}(t) + \underline{V}_{thF}(t) \end{array}\right] \tag{G.6}$$

Here,

$$\mathbf{Z_{th}} = \mathbf{P_S}^T\mathbf{Y_S}^{-1}\mathbf{P_S} + \mathbf{P_F}^T\mathbf{Y_F}^{-1}\mathbf{P_F} \tag{G.7}$$

$$\underline{V}_{thS}(t) = -\mathbf{P_S}^T\mathbf{Y_S}^{-1}\mathbf{A_{iS}}^T\,\underline{h}_{bS}(t) \tag{G.8}$$

$$\underline{V}_{thF}(t) = -\mathbf{P_F}^T\mathbf{Y_F}^{-1}\mathbf{A_{iF}}^T\,\underline{h}_{bF}(t) \tag{G.9}$$

Using the 2^{nd} and the 3^{rd} block equation in (G.6), we can get the following:

$$\underline{i}_\alpha(t) = \mathbf{Z_{th}}^{-1}\left(\underline{V}_{thS}(t) + \underline{V}_{thF}(t)\right) \tag{G.10}$$

$$\underline{v}_F(t) = -\mathbf{Y_F}^{-1}\mathbf{A_{iF}}^T\,\underline{h}_{bF}(t) - \mathbf{Y_F}^{-1}\mathbf{P_F}\,\underline{i}_\alpha(t) \tag{G.11}$$

$$\tag{G.12}$$

Pre-multiplying (G.4) by $\mathbf{Y_S}^{-1}$ and re-arranging the gives (G.13).

$$\underline{v}_S(t) = -\mathbf{Y_S}^{-1}\mathbf{A_{iS}}^T\,\underline{h}_{bS}(t) - \mathbf{Y_S}^{-1}\mathbf{P_S}\left\{\frac{1}{N}\sum_{r=1}^{N}\underline{i}_\alpha(t - \Delta T + r\Delta t)\right\} \tag{G.13}$$

We can see that (G.8) - (G.13) are exactly same as (5.31) - (5.35). Thus, this shows that (5.31) - (5.35) are equivalent to (5.39) - (5.42).

G.2 Proof that (5.44) - (5.47) are equivalent to (5.49) - (5.52)

We start with (5.49) - (5.52). Here, these are repeated in (G.14) - (G.17).

$$\underline{h}_{bS}(t+k\Delta t) = \underline{h}_{bS}(t) + \frac{k}{N}\left(\underline{h}_{bS}(t+\Delta T) - \underline{h}_{bS}(t)\right) \tag{G.14}$$

$$\mathbf{Y_S}\,\underline{v}'_S(t+k\Delta t) + \mathbf{P_S}\,\underline{i}_\alpha(t+k\Delta t) + \mathbf{A_{iS}}^T\,\underline{h}_{bS}(t+k\Delta t) = \underline{0} \tag{G.15}$$

$$\mathbf{Y_F}\,\underline{v}_F(t+k\Delta t) + \mathbf{P_F}\,\underline{i}_\alpha(t+k\Delta t) + \mathbf{A_{iF}}^T\,\underline{h}_{bF}(t+k\Delta t) = \underline{0} \tag{G.16}$$

$$\mathbf{P_S}^T\underline{v}'_S(t+k\Delta t) + \mathbf{P_F}^T\underline{v}_F(t+k\Delta t) = \underline{0} \tag{G.17}$$

Arranging (G.15) - (G.17) in a matrix form yields (G.18).

$$\left[\begin{array}{c|c|c} \mathbf{Y_S} & \mathbf{0} & \mathbf{P_S} \\ \hline \mathbf{0} & \mathbf{Y_F} & \mathbf{P_F} \\ \hline \mathbf{P_S}^T & \mathbf{P_F}^T & \mathbf{0} \end{array}\right]\left[\begin{array}{c} \underline{v}'_S(t+k\Delta t) \\ \hline \underline{v}_F(t+k\Delta t) \\ \hline \underline{i}_\alpha(t+k\Delta t) \end{array}\right] = \left[\begin{array}{c} -\mathbf{A_{iS}}^T\underline{h}_{bS}(t+k\Delta t) \\ \hline -\mathbf{A_{iF}}^T\underline{h}_{bF}(t+k\Delta t) \\ \hline 0 \end{array}\right] \tag{G.18}$$

By performing a few elementary matrix manipulations, (G.18) can be transformed to (G.19).

$$\left[\begin{array}{c|c|c} \mathbf{I_S} & \mathbf{0} & \mathbf{Y_S}^{-1}\mathbf{P_S} \\ \hline \mathbf{0} & \mathbf{I_F} & \mathbf{Y_F}^{-1}\mathbf{P_F} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{Z_{th}} \end{array}\right]\left[\begin{array}{c} \underline{v}'_S(t+k\Delta t) \\ \hline \underline{v}_F(t+k\Delta t) \\ \hline \underline{i}_\alpha(t+k\Delta t) \end{array}\right] = \left[\begin{array}{c} -\mathbf{Y_S}^{-1}\mathbf{A_{iS}}^T\underline{h}_{bS}(t+k\Delta t) \\ \hline -\mathbf{Y_F}^{-1}\mathbf{A_{iF}}^T\underline{h}_{bF}(t+k\Delta t) \\ \hline \underline{V}_{thS}(t+k\Delta t) + \underline{V}_{thF}(t+k\Delta t) \end{array}\right] \tag{G.19}$$

Here,

$$\mathbf{Z_{th}} = \mathbf{P_S}^T\mathbf{Y_S}^{-1}\mathbf{P_S} + \mathbf{P_F}^T\mathbf{Y_F}^{-1}\mathbf{P_F} \tag{G.20}$$

$$\underline{V}_{thS}(t+k\Delta t) = -\mathbf{P_S}^T\mathbf{Y_S}^{-1}\mathbf{A_{iS}}^T\,\underline{h}_{bS}(t+k\Delta t) \tag{G.21}$$

$$\underline{V}_{thF}(t+k\Delta t) = -\mathbf{P_F}^T\mathbf{Y_F}^{-1}\mathbf{A_{iF}}^T\,\underline{h}_{bF}(t+k\Delta t) \tag{G.22}$$

Substituting the value of $\underline{h}_{bS}(t+k\Delta t)$ from (G.14) into (G.21) yields (G.23).

$$\begin{aligned}\underline{V}_{thS}(t+k\Delta t) &= -\mathbf{P_S}^T\mathbf{Y_S}^{-1}\mathbf{A_{iS}}^T\left\{\underline{h}_{bS}(t)+\frac{k}{N}\left(\underline{h}_{bS}(t+\Delta T)-\underline{h}_{bS}(t)\right)\right\}\\ &= \underline{V}_{thS}(t)+\frac{k}{N}\left(\underline{V}_{thS}(t+\Delta T)-\underline{V}_{thS}(t)\right)\end{aligned} \tag{G.23}$$

Using the 2^{nd} and the 3^{rd} block equation in (G.19), we can get the following:

$$\underline{i}_{\alpha}(t+k\Delta t) = \mathbf{Z_{th}}^{-1}\left(\underline{V}_{thS}(t)+\underline{V}_{thF}(t)\right) \tag{G.24}$$

$$\underline{v}_{F}(t+k\Delta t) = -\mathbf{Y_F}^{-1}\mathbf{A_{iF}}^T\,\underline{h}_{bF}(t+k\Delta t)-\mathbf{Y_F}^{-1}\mathbf{P_F}\,\underline{i}_{\alpha}(t+k\Delta t) \tag{G.25}$$

We can see that (G.22) - (G.25) are exactly same as (5.44) - (5.47). Thus, this shows that (5.44) - (5.47) are equivalent to (5.49) - (5.52).

References

[1] J. A. Hollman and J. R. Marti, "Step-by-step eigenvalue analysis with emtp discrete-time solutions," *IEEE Transactions on Power Systems*, vol. 25, no. 3, pp. 1220–1231, Feb. 2010.

[2] F. Moreira and J. Marti, "Latency techniques for time-domain power system transients simulation," *IEEE Transactions on Power Systems*, vol. 20, no. 1, pp. 246–253, Feb. 2005.

[3] T. Noda and S. Sasaki, "Algorithms for Distributed Computation of Electromagnetic Transients toward PC Cluster based Real-time Simulations," in *Proceedings of the International Conference on Power Systems Transients (IPST 2003), New Orleans, USA*, 2003.

[4] J. Bélanger, P. Venne, and J.-N. Paquin, "The what, where and why of real-time simulation," *Planet Rt*, vol. 1, no. 1, pp. 25–29, 2010.

[5] C. J. Truax, J. D. Brown, and W. Neugebauer, "TNA Study of Reclosing Transients on a 765 kV Shunt Compensated Transmission Line," *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-97, no. 4, pp. 1447–1457, Jul. 1978.

[6] W. Ren, "Accuracy evaluation of Power Hardware-in-the-Loop (PHIL) simulation," Ph.D. dissertation, The Florida State University, 2007.

[7] P. W. Sauer, M. A. Pai, and J. H. Chow, *Power System Dynamics and Stability: with Synchrophasor Measurement and Power System Toolbox*. John Wiley & Sons, 2017.

[8] N. Watson and J. Arrillaga, *Power Systems Electromagnetic Transients Simulation*. IET, 2003, vol. 39.

[9] H. W. Dommel, "Digital computer solution of electromagnetic transients in single-and multiphase networks," *IEEE Transactions on Power Apparatus and Systems*, no. 4, pp. 388–399, Apr. 1969.

[10] C.-W. Ho, A. Ruehli, and P. Brennan, "The modified nodal approach to network analysis," *IEEE Transactions on Circuits and Systems*, vol. 22, no. 6, pp. 504–509, Jun. 1975.

[11] G. Strang, *Introduction to Linear Algebra*. Wellesley-Cambridge Press Wellesley, MA, 1993, vol. 3.

[12] T. Maguire and J. Giesbrecht, "Small time-step (< 2μSec) VSC model for the Real Time Digital Simulator," in *International Conference on Power System Transients (IPST) 2005, Montréal, Canada*, 2005.

[13] G. Anderson, N. Watson, N. Arnold, and J. Arrillaga, "A new hybrid algorithm for analysis of HVDC and FACTS systems," in *Proceedings 1995 International Conference on Energy Management and Power Delivery EMPD '95*, vol. 2, 1995, pp. 462–467.

[14] M. Heffernan, K. Turner, J. Arrillaga, and C. Arnold, "Computation of A.C.-D.C. System Disturbances - Part I. Interactive Coordination of Generator and Convertor Transient Models," *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-100, no. 11, pp. 4341–4348, Nov. 1981.

[15] K. Mudunkotuwa, S. Filizadeh, and U. Annakkage, "Development of a Hybrid Simulator by interfacing Dynamic Phasors with Electromagnetic Transient Simulation," *IET Generation, Transmission \& Distribution*, vol. 11, no. 12, pp. 2991–3001, Sept. 2017.

[16] D. Shu, X. Xie, V. Dinavahi, C. Zhang, X. Ye, and Q. Jiang, "Dynamic Phasor Based Interface Model for EMT and Transient Stability Hybrid Simulations," *IEEE Transactions on Power Systems*, vol. 33, no. 4, pp. 3930–3939, Jul. 2018.

[17] U. D. Annakkage, N. K. C. Nair, Y. Liang, A. M. Gole, V. Dinavahi, B. Gustavsen, T. Noda, H. Ghasemi, A. Monti, M. Matar, R. Iravani, and J. A. Martinez, "Dynamic System Equivalents: A Survey of Available Techniques," *IEEE Transactions on Power Delivery*, vol. 27, no. 1, pp. 411–420, Jan. 2012.

[18] B. Gustavsen and A. Semlyen, "Rational approximation of frequency domain responses by vector fitting," *IEEE Transactions on Power Delivery*, vol. 14, no. 3, pp. 1052–1061, Jul. 1999.

[19] M. Ahmadi, S. Fan, A. M. Gole, and H. J. De Silva, "A guaranteed passive model for multi-port frequency dependent network equivalents using network synthesis approach," *Electric Power Systems Research*, vol. 197, p. 107248, Aug. 2021.

[20] M. Abdel-Rahman, A. Semlyen, and M. Reza Iravani, "Two-layer network equivalent for electromagnetic transients," *IEEE Transactions on Power Delivery*, vol. 18, no. 4, pp. 1328–1335, Oct. 2003.

[21] X. Nie, Y. Chen, and V. Dinavahi, "Real-time transient simulation based on a robust two-layer network equivalent," *IEEE Transactions on Power Systems*, vol. 22, no. 4, pp. 1771–1781, Nov. 2007.

[22] R. J. Newell, M. D. Risan, L. Allen, I. S. Rao, and D. L. Stuehm, "Utility experience with coherency-based dynamic equivalents of very large systems," *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-104, no. 11, pp. 3056–3063, Nov. 1985.

[23] S. de Oliveira and A. Massaud, "Modal dynamic equivalent for electric power systems. ii. stability simulation tests," *IEEE Transactions on Power Systems*, vol. 3, no. 4, pp. 1731–1737, Nov. 1988.

[24] X. Lin, A. M. Gole, and M. Yu, "A wide-band multi-port system equivalent for real-time digital power system simulators," *IEEE Transactions on Power Systems*, vol. 24, no. 1, pp. 237–249, 2009.

[25] B. Gustavsen and O. Mo, "Interfacing convolution based linear models to an electromagnetic transients program," in *International Conference on Power System Transients (IPST) 2007, Lyon, France*, 2007.

[26] Y. Zhang, A. M. Gole, W. Wu, B. Zhang, and H. Sun, "Development and analysis of applicability of a hybrid transient simulation platform combining tsa and emt elements," *IEEE Transactions on Power Systems*, vol. 28, no. 1, pp. 357–366, Feb. 2013.

[27] B. Gustavsen and A. Semlyen, "Enforcing passivity for admittance matrices approximated by rational functions," *IEEE Transactions on Power Systems*, vol. 16, no. 1, pp. 97–104, Feb. 2001.

[28] Y. Hu, W. Wu, A. M. Gole, and B. Zhang, "A guaranteed and efficient method to enforce passivity of frequency-dependent network equivalents," *IEEE Transactions on Power Systems*, vol. 32, no. 3, pp. 2455–2463, May 2017.

[29] D. M. Falcao, E. Kaszkurewicz, and H. L. Almeida, "Application of parallel processing techniques to the simulation of power system electromagnetic transients," *IEEE Transactions on Power Systems*, vol. 8, no. 1, pp. 90–96, 1993.

[30] J. R. Marti, L. R. Linares, J. Calvino, H. W. Dommel, and J. Lin, "OVNI: An Object Approach to Real-time Power System Simulators," in *POWERCON'98. 1998 International Conference on Power System Technology. Proceedings (Cat. No. 98EX151)*, vol. 2. Beijing, China: IEEE, 1998, pp. 977–981.

[31] D. B. Kirk and W.-M. W. Hwu, *Programming Massively Parallel Processors: A Hands-on Approach*. Morgan Kaufmann, 2016.

[32] R. Singh, A. Gole, P. Graham, J. Muller, R. Jayasinghe, B. Jayasekera, and D. Muthumuni, "Using local grid and multi-core computing in electromagnetic transients simulation," in *International Conference on Power System Transients (IPST) 2013, Vancouver, Canada*, 2013.

[33] S. Montplaisir-Goncalves, J. Mahseredjian, O. Saad, X. Legrand, and A. El-Akoum, "A semaphore-based parallelization of networks for electromagnetic transients," in *International Conference on Power System Transients (IPST) 2015, Cavtat, Croatia*, 2013.

[34] S. Fan, H. Ding, A. Kariyawasam, and A. M. Gole, "Parallel Electromagnetic Transients Simulation with Shared Memory Architecture Computers," *IEEE Transactions on Power Delivery*, vol. 33, no. 1, pp. 239–247, Feb. 2018.

[35] J. K. Debnath, A. M. Gole, and W.-K. Fung, "Graphics-Processing-Unit-based Acceleration of Electromagnetic Transients Simulation," *IEEE Transactions on Power Delivery*, vol. 31, no. 5, pp. 2036–2044, Oct. 2016.

[36] B. Wilkinson and M. Allen, *Parallel Programming*. Pearson India, 2004.

[37] J. R. Marti and L. R. Linares, "Real-time EMTP-based Transients Simulation," *IEEE Transactions on Power Systems*, vol. 9, no. 3, pp. 1309–1317, Aug. 1994.

[38] J. M. Zavahir, J. Arrillaga, and N. R. Watson, "Hybrid Electromagnetic Transient Simulation with the State Variable Representation of HVDC Converter Plant," *IEEE transactions on Power Delivery*, vol. 8, no. 3, pp. 1591–1598, Jul. 1993.

[39] S. Y. R. Hui, K. K. Fung, and C. Christopoulos, "Decoupled Simulation of DC-linked Power Electronic Systems using Transmission-line Links," *IEEE Transactions on Power Electronics*, vol. 9, no. 1, pp. 85–91, Jan. 1994.

[40] J. Bélanger, L. A. Snider, J.-N. Paquin, C. Pirolli, and W. Li, "A Modern and Open Real-time Digital Simulator of Contemporary Power Systems," in *Proceedings of the International Conference on Power Systems Transients (IPST 2009), Kyoto, Japan*, 2009, pp. 2–6.

[41] G. Kron, *Diakoptics : Piecewise Solutions of Large Systems*. MacDonald Publishing, London, 1963, vol. 1.

[42] K. Strunz and E. Carlson, "Nested Fast and Simultaneous Solution for Time-domain Simulation of Integrative Power-electric and Electronic Systems," *IEEE Transactions on Power Delivery*, vol. 22, no. 1, pp. 277–287, Jan. 2007.

[43] C. Dufour, J. Mahseredjian, and J. Bélanger, "A Combined State-Space Nodal Method for the Simulation of Power System Transients," *IEEE Transactions on Power Delivery*, vol. 26, no. 2, pp. 928–935, Apr. 2011.

[44] G. Karypis and V. Kumar, "Multilevelk-way partitioning scheme for irregular graphs," *Journal of Parallel and Distributed computing*, vol. 48, no. 1, pp. 96–129, Jan. 1998.

[45] I. S. Duff and J. K. Reid, "Algorithm 529: Permutations to block triangular form [f1]," *ACM Transactions on Mathematical Software (TOMS)*, vol. 4, no. 2, pp. 189–192, June 1978.

[46] R. Yonezawa and T. Noda, "A Study of Solution Process Parallelization for an EMT Analysis Program Using OpenMP," in *International Conference on Power System Transients (IPST) 2017, Seoul, Republic of Korea*, 2017.

[47] A. Abusalah, O. Saad, J. Mahseredjian, U. Karaagac, and I. Kocar, "Accelerated Sparse Matrix-Based Computation of Electromagnetic Transients," *IEEE Open Access Journal of Power and Energy*, vol. 7, pp. 13–21, Jan. 2020.

[48] T. Duan and V. Dinavahi, "A novel linking-domain extraction decomposition method for parallel electromagnetic transient simulation of large-scale ac/dc networks," *IEEE Transactions on Power Delivery*, vol. 36, no. 2, pp. 957–965, Apr. 2021.

[49] C. F. Van Loan and G. H. Golub, *Matrix Computations*. Johns Hopkins University Press Baltimore, 1983.

[50] A. Semlyen and F. de Leon, "Computation of electromagnetic transients using dual or multiple time steps," *IEEE Transactions on Power Systems*, vol. 8, no. 3, pp. 1274–1281, Aug. 1993.

[51] L. R. Linares and J. R. Marti, "Sub-Area Latency in a Real Time Power Network Simulator," in *International Conference on Power System Transients (IPST) 1995, Lisbon, Portugal*, 1995.

[52] *VSC Small Time-Step Modelling Manual*, RTDS Technologies Inc., Winnipeg, MB, Canada, 2006.

[53] S. Pekarek, O. Wasynczuk, E. Walters, J. Jatskevich, C. Lucas, N. Wu, and P. Lamm, "An efficient multirate simulation technique for power-electronic-based systems," *IEEE Transactions on Power Systems*, vol. 19, no. 1, pp. 399–409, Feb. 2004.

[54] A. Benigni, A. Monti, and R. A. Dougal, "Latency-Based Approach to the Simulation of Large Power Electronics Systems," *IEEE Transactions on Power Electronics*, vol. 29, no. 6, pp. 3201–3213, June 2014.

[55] F. A. Moreira, J. R. Marti, L. C. Zanetta, and L. R. Linares, "Multirate Simulations With Simultaneous-Solution Using Direct Integration Methods in a Partitioned Network Environment," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 53, no. 12, pp. 2765–2778, Dec. 2006.

[56] T. Chen and B. A. Francis, *Optimal Sampled-data Control Systems*. Springer Science & Business Media, 2012.

[57] D. A. Woodford, A. M. Gole, and R. W. Menzies, "Digital simulation of dc links and ac machines," *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-102, no. 6, pp. 1616–1623, Jun. 1983.

[58] R. P. Wierckx, W. J. Giebrecht, R. Kuffel, X. Wang, G. B. Mazur, M. A. Weekes, and A. M. Gole, "Validation of a fully digital real-time electromagnetic transients simulator for hvdc system controls studies," in *Proceedings. Joint International Power Conference Athens Power Tech,*, vol. 2, 1993, pp. 751–759.

[59] J. Mahseredjian, S. Dennetière, L. Dubé, B. Khodabakhchian, and L. Gérin-Lajoie, "On a new approach for the simulation of transients in power systems," *Electric Power Systems Research*, vol. 77, no. 11, pp. 1514–1520, Sept. 2007.

[60] S. Seshu and N. Balabanian, *Linear Network Analysis*. Wiley, 1959.

[61] S. Natarajan, "A systematic method for obtaining state equations using mna," *IEE Proceedings G-Circuits, Devices and Systems*, vol. 138, no. 3, pp. 341–346, Jun. 1991.

[62] A.-R. Sana, J. Mahseredjian, X. Dai-Do, and H. Dommel, "Treatment of discontinuities in time-domain simulation of switched networks," *Mathematics and computers in simulation*, vol. 38, no. 4-6, pp. 377–387, Aug. 1995.

[63] H. Zhao, S. Fan, and A. Gole, "Equivalency of state space models and emt companion circuit models," in *International Conference on Power System Transients (IPST) 2019, Perpinan, France*, 2019.

[64] H. W. Dommel, *Electromagnetic Transients Program (EMTP) Theory Book*. Bonneville Power Administration, 1986.

[65] H. Zhao, S. Fan, and A. M. Gole, "Stability of Algorithms for Electro-Magnetic-Transient Simulation of Networks with Switches and Non-linear Inductors," *IEEE Transactions on Power Delivery*, vol. 35, no. 1, pp. 377–385, Feb. 2020.

[66] Q. Chen, S.-H. Weng, and C.-K. Cheng, "A Practical Regularization Technique for Modified Nodal Analysis in Large-scale Time-domain Circuit Simulation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, no. 7, pp. 1031–1040, Jul. 2012.

[67] A. B. Birchfield, T. Xu, K. M. Gegner, K. S. Shetye, and T. J. Overbye, "Grid structural characteristics as validation criteria for synthetic networks," *IEEE Transactions on Power Systems*, vol. 32, no. 4, pp. 3258–3265, Jul. 2017.

[68] "Electric Grid Test Cases - Texas A&M University," online; accessed 16 September 2021. [Online]. Available: https://electricgrids.engr.tamu.edu/electric-grid-test-cases/

[69] S. M. Chan and V. Brandwajn, "Partial Matrix Refactorization," *IEEE Transactions on Power Systems*, vol. 1, no. 1, pp. 193–199, Feb. 1986.

[70] H. W. Dommel, "Nonlinear and Time-Varying Elements in Digital Simulation of Electromagnetic Transients," *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-90, no. 6, pp. 2561–2567, Nov. 1971.

[71] *PSCAD User Manual v4.6.0*, Manitoba HVDC Research Centre, Winnipeg, MB, Canada, 2018.

[72] "PSCAD Knowledge Base - Models and Examples (IEEE Test Systems)," Manitoba HVDC Research Centre, Winnipeg, Canada, online; accessed 22 July 2021. [Online]. Available: https://www.pscad.com/knowledge-base/topic-51/v-

[73] M. Szechtman, "First benchmark model for hvdc control studies," *Electra*, vol. 135, pp. 55–73, 1991.

[74] R. Lasseter and K. Kruger, "HVDC simulation using NETOMAC," in *IEEE Montech'86, Conference on HVDC Power Transmision*, 1986.

[75] A. M. Gole, I. T. Fernando, G. D. Irwin, and O. B. Nayak, "Modeling of Power Electronic Apparatus: Additional Interpolation Issues," in *Proceedings of the International Conference on Power Systems Transients (IPST 1997), Seattle, USA*, 1997.

[76] G. D. Irwin, D. A. Woodford, and A. M. Gole, "Precision Simulation Of PWM Controllers," in *International Conference on Power System Transients (IPST) 2001, Rio de Janeiro, Brazil*, 2001.

[77] "The OpenMP API specification for parallel programming," online; accessed 28 Sept. 2022. [Online]. Available: https://www.openmp.org/

[78] T. G. Mattson, Y. H. He, and A. E. Koniges, *The OpenMP Common Core: Making OpenMP Simple Again*. MIT Press, 2019.

[79] T. Tessem, *Improving Parallel Sparse Matrix-vector Multiplication*. M.Sc Thesis, Department of Informatics, University of Bergen, December 2013.

[80] T. A. Davis, *Direct Methods for Sparse Linear Systems*. SIAM, 2006.

[81] P. R. Amestoy, T. A. Davis, and I. S. Duff, "An approximate minimum degree ordering algorithm," *SIAM Journal on Matrix Analysis and Applications*, vol. 17, no. 4, pp. 886–905, 1996.

[82] "Suitesparse - A Suite of Sparse Matrix Software," online; accessed 05 October 2022. [Online]. Available: https://people.engr.tamu.edu/davis/suitesparse.html

[83] "METIS - Serial Graph Partitioning and Fill-reducing Matrix Ordering," online; accessed 02 October 2021. [Online]. Available: http://glaros.dtc.umn.edu/gkhome/metis/metis/overview

[84] A. M. Gole, "Modelling of Transmission Lines - ECE 7310 Power System Transient Simulation," Course Notes; accessed 10 November 2021. [Online]. Available: http://home.cc.umanitoba.ca/~gole/ECE_7310/notes_files/lecture4-1.pdf

[85] R. Kuffel, J. Giesbrecht, T. Maguire, R. Wierckx, and P. McLaren, "Rtds-a fully digital power system simulator operating in real time," in *Proceedings 1995 International Conference on Energy Management and Power Delivery EMPD '95*, vol. 2, 1995, pp. 498–503 vol.2.

[86] "HYPERSIM User Documentation," OPAL-RT Technologies, online; accessed 22 July 2021. [Online]. Available: https://wiki.opal-rt.com/display/DOCHS/HYPERSIM+User+Documentation

[87] P. Triverio, S. Grivet-Talocia, M. S. Nakhla, F. G. Canavero, and R. Achar, "Stability, Causality, and Passivity in Electrical Interconnect Models," *IEEE Transactions on Advanced Packaging*, vol. 30, no. 4, pp. 795–808, Nov. 2007.

[88] J. Mahseredjian and F. Alvarado, "Creating an Electromagnetic Transients Program in MATLAB: MatEMTP," *IEEE Transactions on Power Delivery*, vol. 12, no. 1, pp. 380–388, Jan. 1997.

[89] J. R. Marti and J. Lin, "Suppression of numerical oscillations in the EMTP power systems," *IEEE Transactions on Power Systems*, vol. 4, no. 2, pp. 739–747, May 1989.

[90] L. V. Bewley, "Traveling waves on transmission systems," *Transactions of the American Institute of Electrical Engineers*, vol. 50, no. 2, pp. 532–550, jun 1931.

[91] A. Morched, B. Gustavsen, and M. Tartibi, "A universal model for accurate calculation of electromagnetic transients on overhead lines and underground cables," *IEEE Transactions on Power Delivery*, vol. 14, no. 3, pp. 1032–1038, Jul. 1999.

[92] "Guide for the Development of Models for HVDC Converters in a HVDC Grid," CIGRE WG B4-57 Technical Brochure. [Online]. Available: https://e-cigre.org/publication/604-guide-for-the-development-of-models-for-hvdc-converters-in-a-hvdc-gridJuly22, 2021]

[93] A. Yazdani and R. Iravani, *Voltage-sourced Converters in Power Systems: Modeling, Control, and Applications*. John Wiley & Sons, 2010.

[94] Y. Shu, G. Tang, and H. Pang, "A back-to-back VSC-HVDC system of Yu-E power transmission lines to improve cross-region capacity," *CSEE Journal of Power and Energy Systems*, vol. 6, no. 1, pp. 64–71, Mar. 2019.

[95] J. Peralta, H. Saad, S. Dennetiere, J. Mahseredjian, and S. Nguefeu, "Detailed and Averaged Models for a 401-Level MMC–HVDC System," *IEEE Transactions on Power Delivery*, vol. 27, no. 3, pp. 1501–1508, Jul. 2012.

[96] B. Jacobson, P. Karlsson, G. Asplund, L. Harnefors, and T. Jonsson, "VSC-HVDC transmission with cascaded two-level converters," in *Cigré session*, Paris, 2010, pp. B4–B110.

[97] S. Kouro, R. Bernal, H. Miranda, C. A. Silva, and J. Rodriguez, "High-Performance Torque and Flux Control for Multilevel Inverter Fed Induction Motors," *IEEE Transactions on Power Electronics*, vol. 22, no. 6, pp. 2116–2123, Nov. 2007.

[98] J. Arrillaga, *High Voltage Direct Current Transmission*. IET, 1998, vol. 29.

[99] A. M. Gole and V. K. Sood, "A static compensator model for use with electromagnetic transients simulation programs," *IEEE Transactions on Power Delivery*, vol. 5, no. 3, pp. 1398–1407, Jul. 1990.

[100] M. L. Crow and J. G. Chen, "The multirate simulation of FACTS devices in power system dynamics," *IEEE Transactions on Power Systems*, vol. 11, no. 1, pp. 376–382, Feb. 1996.

[101] L.-A. Grégoire, H. F. Blanchette, J. Bélanger, and K. Al-Haddad, "A stability and accuracy validation method for multirate digital simulation," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 2, pp. 512–519, Apr. 2017.

[102] A. Iserles, *A First Course in the Numerical Analysis of Differential Equations*. Cambridge University Press, 1996.

[103] G. Michaletzky and L. Gerencsér, "BIBO Stability of Linear Switching Systems," *IEEE Transactions on Automatic Control*, vol. 47, no. 11, pp. 1895–1898, 2002.

[104] V. Mehrmann and T. Stykel, "Descriptor systems: A general mathematical framework for modelling, simulation and control," *Automatisierungstechnik*, vol. 54, no. 8, pp. 405–415, 2006.

[105] MATLAB 9.9.0.1524773 (R2020b). Natick, Massachusetts: The MathWorks Inc., 2020.