SPICE Compatible Macromodels of Linear Multiport Networks

by

Liu Zhaoqing

Department of Electrical& Computer Engineering

McGill University, Montreal, Canada

April 2015

A thesis submitted to McGill University in partial fulfillment of the requirements of the degree of Master of Engineering

Abstract

In the past few decades, complexity and operating frequency of microelectronic circuits have considerably increased and their sizes have shrunk. As a result, EDA simulations for complex systems have become difficult tasks for two main reasons. One is that their actual physical models are, sometimes, impossible to obtain. The other reason is that some characteristics of component are modified under high-frequency operation. In this case, the bottleneck problem of macromodeling is to develop the accurate analytical model under high frequency environment. To solve the problems for passive linear systems, frequency-domain S or Y-parameter data can be obtained at first through measurements or full-wave simulation.

To deal with the frequency-domain measurements, fitting methods (Vector Fitting or Loewner Matrix method) are applied to obtain the mathematical macromodel. They all can handle high complexity system, but generate different macromodels for the results. For example, VF method generates results in pole-residue approximation model; and LM method generates results in state space matrices model.

From these macromodels above, this thesis proposes several approaches to derive SPICE level simulation model (or SPICE netlist) when inputting measurements for fitting methods are Y-parameter. SPICE netlist is in format of components list (resistors, capacitors, inductors and etc.); so it is convenient for EDA simulation. The ideas of these approaches are based on properties of MNA equation and reactance theorem. They are proven to generate accurate results and they can greatly accelerate the macromodeling process in this thesis.

This thesis also provides an open discussion on SPICE netlist generation from frequency-domain data. Within the development of new or optimized fitting methods, we always can invent or optimize the corresponding macromodeling methods to better adapt characteristics of those fitting methods. On the other hand, the best solution for different problems (or systems), is not fixed. Adopting the method based on the properties of the system would achieve better efficiency.

Résumé

Au cours des dernières décennies, la complexité et la fréquence de fonctionnement des circuits microélectroniques ont considérablement augmenté et leurs tailles ont diminué. En conséquence, les simulations d'ACE (l'automatisation de la conception électronique) de systèmes complexes sont devenues des tâches difficiles pour deux raisons principales. La première raison est que leurs modèles physiques réels étaient impossibles à obtenir parfois. L'autre raison est que certaines caractéristiques du composant étaient modifiées en fonctionnement à haute fréquence. Dans ce cas, le problème de la macro- modélisation est le développement le modèle d'une analytique précise dans un environnement à haute fréquence. Pour résoudre les problèmes pour les systèmes linéaires passifs, S ou Y-paramètre dans le domaine fréquentiel peuvent être obtenus premièrement par des mesures ou par la simulation d'onde-entière.

Pour traiter les mesures de domaine de fréquence, les méthodes d'ajustement (méthodes "Vector Fitting " ou "Loewner Matrix") sont appliquées pour obtenir la macro-modèle mathématique. Tous peuvent gérer les systèmes de complexité élevée, mais ils génèrent macro-modèles différentes pour les résultats. Par exemple, la méthode VF génère des résultats dans le modèle d'approximation de pôle-résidus; et la méthode LM génère des résultats dans le modèle de matrices de l'espace d'état.

Selon ces macro-modèles, cette thèse propose plusieurs approches pour tirer modèle de la simulation au niveau SPICE (ou la netliste SPICE) lorsque les mesures d'entrée pour des méthodes d'ajustement sont Y-paramètre. La netliste SPICE est au format de la liste des composants (résistances, condensateurs, inductances et etc.); il est si commode pour la simulation de l'ACE. Les idées de ces approches sont basées sur les caractéristiques de l'équation MNA et sur le théorème de la réactance. Ils sont prouvés de générer des résultats précis et ils peuvent considérablement accélérer le processus de la macro-modélisation dans cette thèse.

Cette thèse fournit également une discussion ouverte sur la génération de la netlist SPICE à des données de domaine de fréquence. Dans le développement de nouvelles méthodes d'ajustement, nous pouvons toujours inventer ou optimiser les méthodes de macro-modélisation afin de mieux adapter les caractéristiques de ces méthodes d'ajustement. De l'autre côté, la meilleure solution pour différents problèmes (ou systèmes), n'est pas fixe. L'adoption de la méthode sur la base des caractéristiques du système pourrait atteindre une meilleure efficacité.

Acknowledgements

There are so many people I would like to thank for completing this thesis. First and foremost, I would like to express my deepest and sincerest appreciation to my supervisor, Professor Roni Khazaka. Without his guidance, support, patience and encouragement during my master study, I couldn't accomplish this thesis. In addition, his extensive knowledge and brand new ideas always impressed me and offered me great passion on researches. As a result, I had enjoyed my three-year master study in McGill University.

I also would like to thank my colleagues for the assistances on my studies and researches. Marwan Kanaan helped my setting up the lab workstation and provided maintenances on computer and printer. During my study of course 'Circuit Simulation', he was the TA and offered me assistances on Matlab programming. Muhammad Kabir provided me much knowledge on Loewner Matrix method and taught me ANSYS and HFSS. He also provided me the code for LM method and two examples for simulation. I also would like to thank all my other colleges, Julian Santorelli, Nassir AbouZiki and Macro Kassis, for sharing many presentations and having discussions on our research paths. I would like to thank my other friends in lab McConnell 528, they are Chuansheng Dong, Hangcheng Zhu, Xingchi Chen and Di Wu. Although we were working for different subjects, we always shared the knowledge and understanding in field of Electrical Engineering. I would like to thank all staffs in Electrical Engineering Department of McGill University for the accommodation.

Furthermore, I appreciate the internship opportunities offered from Canada Royalties Inc and China Railway Siyuan Survery and Design Group. These experiences would be necessary for my future career.

Finally, I must give greatest appreciation to my parents, Liu Jiancheng and Xie Zhuoyan, for all the loves, guidance and supports during my life so far. Without their encouragements and generous supports, I wouldn't come to Montreal and study at here. All my successes are established on their patience and devotement.

Table of Contents

List of Figures

List of Tables

List of Acronyms

EDA	Electronic Design Automation
MEMS	Microelectromechanical Systems
MNA	Modified nodal analysis
MOR	Model order reduction
SPICE	Simulation Program with Integrated Circuit Emphasis
VF	Vector fitting
LM	Loewner matrix
LS	Least squares
CPU	Central Processing Unit
PRIMA	Passive Reduced-order Interconnect Macromodeling Algorithm
FFT	Fast Fourier Transform
FMM	Fast Matrix-Vector Multiplication
FEM	Finite element method
KCL	Kirchoff's Current Law
PR	positive real
SPR	Strictly positive real
AWE	Asymptotic Waveform Evaluation
VCCS	Voltage controlled current source
ORHP	Open Right Half Plane
SMA	Sparse Matrices Algorithm
MFM	Modified Foster's Method
FBM	Foster-Brune Method

Chapter 1 Introduction

1.1 Background and motivation

In the past few decades, electronic components have become ubiquitous, appearing in diverse systems from computers to cars, smartphones, TV and other small and large appliances. This was partially facilitated by lower costs due to manufacturing process as well as systematic design automation of electronic systems. Such electronic design automation (EDA) tools have therefore been one of the key enablers of the current digital revolution that has seen microelectronics integrated in almost every product. However, at the same time the complexity of microelectronic circuit has considerably increased. It is partially due to increased system complexity due to higher integration and partially due to more complex models required for high frequency operation reaching well into the GHz range. It has put a considerable strain on the EDA tools used for design. In fact EDA tools have now become one of the key bottlenecks limiting the performance and complexity of modern designs [1, 2].

One of the key requirements for accurate simulation and design automation is an accurate and efficient model for the circuit components. However, for many complex components, especially those operate at high frequency, such as transmission lines, Microelectromechanical systems (MEMS), interconnectors and vias, accurate physics based models are not easily obtained. In this case, the bottleneck problem is to develop the accurate analytical time domain models [3] under high frequency environment. However, for passive linear structures, we can fully characterize their behavior by the frequency domain S- or Y-parameter data, which can be obtained from measurement or full-wave simulation. The key challenge is how to include frequency domain data into a linear time domain simulation. This is the main focus of this thesis.

Throughout all present analytical models, SPICE level simulation models [10] show superior behavior for complicated high-frequency circuits or networks. Simulation Program with Integrated Circuit (SPICE) is a typical analog electronic circuit simulator. It directly describes the integrated circuit structures and is convenient for time domain and frequency

domain analysis. In later part of this thesis, this model will be called as 'SPICE netlist', which indicate the virtual components of the circuit and their connection data between nodes in the circuit. This thesis will focus on the circuits or networks which have large number of ports and work under high frequency ($\geq$1GHz).

Macromodeling for high frequency simple components, such as a capacitor, can be achieved through analyzing the physical structures. But for high frequency components, such as transmission lines, physical structures are very hard to obtain. In this case, frequency domain measurements are considered as an assistance of macromodeling. Starting from these measurements, there are usually three approaches to perform a transient time domain simulation: Direct convolution method, Recursive convolution method and SPICE compatible Macromodel method. They will be mentioned in Section 2.1.

1.2 Thesis Contributions

In this thesis, a number of macromodeling methods are developed for frequency measurement data (Y- parameter) of arbitrary system. These methods extend the capability on complicated applications and improve the efficiency on simulation; they are listed as follow.

1. "Sparse Matrices Algorithm" for state-space matrices from Loewner Matrix Method (see Section 4.2): state-space matrices of an arbitrary system could be fast generated from frequency response through SMA method. Since these matrices are usually full matrices, utilizing them efficiently became another issue. Sparse Matrices Algorithm analyzes the properties of these state-space matrices; provides a quick way to obtain the sparsified matrices and then fits them to MNA equation to obtain the SPICE compatible macromodel. As demonstrated in examples, this proposed algorithm significantly reduced the complexity of SPICE compatible model.

2. Modified Foster's Method (see Section 5.3, Appendix II): When Vector Fitting Method is applied on frequency response of an arbitrary system; the results would be in pole-residues format. Chapter 5 mainly discusses algorithms which could generate SPICE compatible models from these pole-residue rational approximations. Modified Foster's Method, proposed in Section 5.3, is a method using Foster-like circuit or Foster's Reactance theorem [58] to derive the virtual components for a frequency response. Several prototypes are acclaimed so that rational approximation of a frequency response can quickly address to different modes. MFM method has very low time cost to generate SPICE compatible macromodel from rational approximations.

3. Foster-Brune Method (see Section 5.5): One drawback of Modified Foster's Method is that the number of components of final result would be massive when the number of gussed poles used in VF method is large. To optimize Modified Foster's Method, reduction for the number of components is the key point. After having reviewed Brune's method, Section 5.5 has proposed "Foster-Brune Method". This method has first established a selecting algorithm on all pole-residues rational approximations, then has extracted, as many as possible, single pole-residue parts and has grouped them into high-order positive-real rational approximations. The next step is to apply Brune's method on these grouped ones and MFM on those ungrouped ones.

4. Positive-real (PR) conditions for pole-residue components (see Appendix IV): For a frequency response function $f(s)$, this function is PR if and only if $Re[f(s)]$ is always positive in the right-half plane of the complex plane and $f(s)$ has real value for real s. This section has considered pole-residue rational function as component, has sought the PR conditions for the sum of any two components. After analysis and proof, all conditions are summarized into format of pseudo code so they could be easily presented in programming.

1.3 Thesis Organization

In Chapter 2, we have introduced three approaches for performing transient analysis for the multiport system at first, and then we have formulated the problem in Section 2.1.In later sections, we have introduced background knowledge, such as direct convolution and various fitting methods. Modified Nodal Analysis and linear subsection are very important for understanding proposed algorithm (SMA) in Chapter 4. They are discussed in section 2.4 and 2.5.

In Chapter 3, we have summarized some fitting techniques for generating time-domain macromodels. They are least squares method (LS)[12], Strictly positive approximation [36-37], Convex programming method [38-39], Vector Fitting method(VF) and Loewner Matrix method(LM). For VF method, we have analyzed basic algorithm on single response, and its application on multiport systems. For introducing LM method, the original method claimed in 2008[54] has been reviewed first. Additionally, several modified methods [47-48, 50], claimed by my supervisor Roni Khazaka and his students, have been reviewed later for providing accurate time-domain models for examples.

Chapter 4 focuses on SPICE netlist generation from state-space matrices. At first, the general method to generate SPICE netlist from state-space matrices have been introduced in Section 4.1. In Section 4.2, I have proposed "**Sparse Matrices Algorithm**", which can generate SPICE netlist from state-space matrices obtained through LM method. This method, utilized the properties of LM method, allows SPICE netlist generation fast and accurate. This method has also already been published in [55], and has been detailedly described in this section.

Chapter 5 focuses on SPICE netlist generation from rational function (or pole-residue rational approximation). Section 5.1 has also reviewed positive-real (PR) property of rational function, which is necessary for generating positive-real component (real world component) in the netlist. Section 5.2 has described the idea to divide Y-parameter into individual component. Section 5.3 has first reviewed Foster-Like circuit [58], which can generate RLC circuits from frequency responses. Then we have made some optimizations and have proposed "**Modified Foster method**" for SPICE netlist generation. Furthermore, we have established several stamps, which allow us to directly generate RLC model from poles-residue sets of multiport network. In Section 5.4, we have reviewed Brune's method [57]; however, this method has some fatal drawbacks on dealing with complex systems.

In Section 5.5, we have combined Brune's method and Modified Foster method and have proposed a new method, "Foster-Brune method", which absorbs the advantages of both methods and avoids their drawbacks.

Chapter 6 is the simulation part of this thesis. Two examples are presented, one is single transmission line and the other one is an 8-port network. We have first applied VF and LM method on these examples to obtain numerical macromodels. Then we have applied Sparse Matrices Algorithm (SMA), Modified Foster method (MFM) and Foster-Brune method (FBM) on these macromodels. The parameters for the comparison between these methods include: accuracy (error), reality (real-world components in netlist) and time consumption. For observing the accuracy for these methods, we have applied Backward Euler method for transient analysis.

Chapter 7 provides the summary of current work. The goal of this thesis is the research on most efficient method generate SPICE compatible model (netlist) from frequency response data. To achieve this goal, some directions of future work are given in the end.

Chapter 2 Problem Formulation and Background Knowledge

As it mentioned in the first chapter, there are three macromodeling methods for dealing with frequency data of an unknown system. After formulating the problem for presenting these methods, it is very necessary to introduce the background knowledge for these methods. They are direct convolution method for Y-parameter, various fitting methods, Modified Nodal Analysis (MNA) and linear subsection.

2.1 General Approaches and Problem Formulation

As it mentioned in end of introduction, there are three approaches to perform a transient time domain simulation from frequency-domain measurements. Their flowcharts are shown in Figure 2.1.

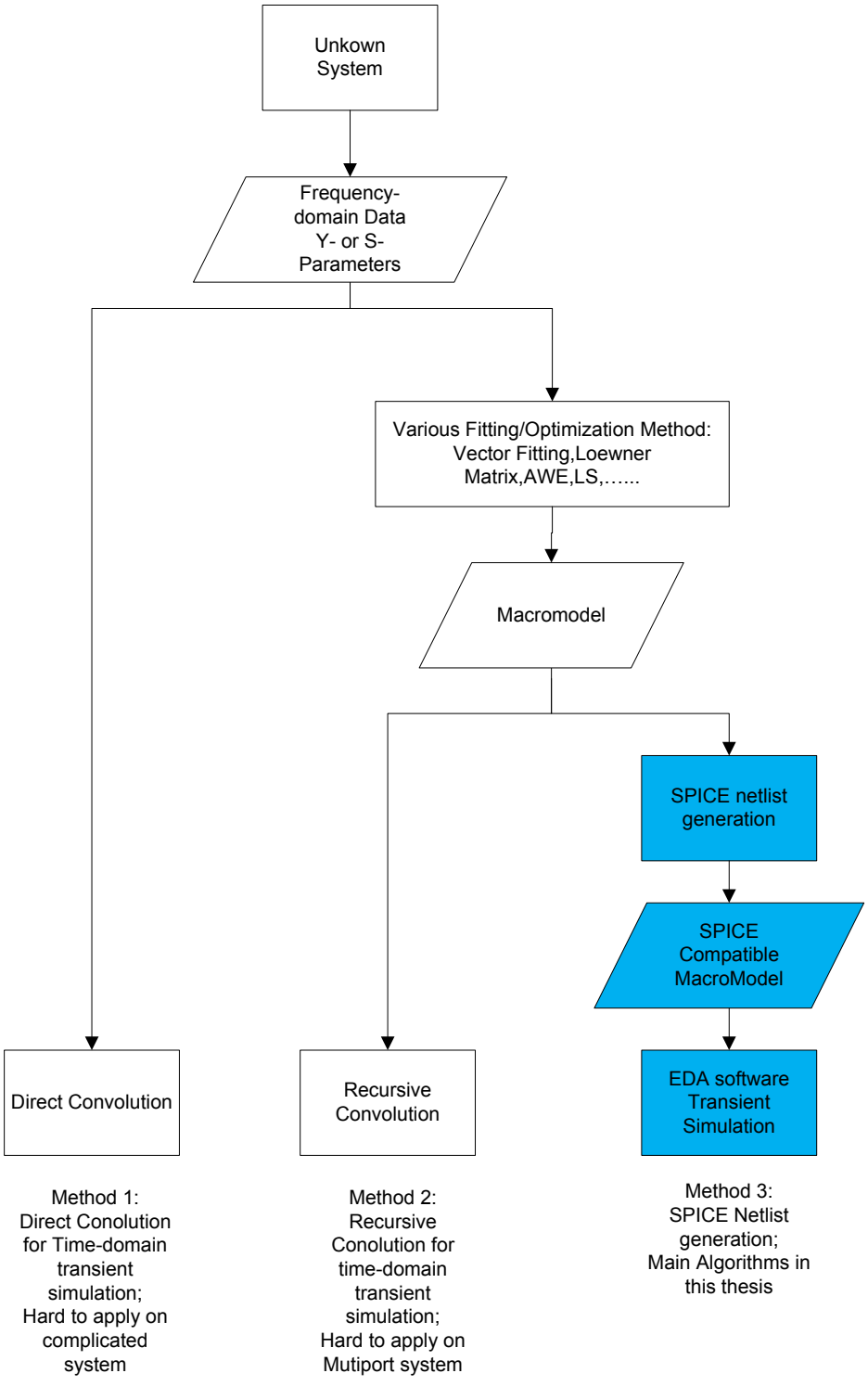


Figure 2.1 Current methods for time-domain transient simulation

1. Direct convolution method can solve time domain models from frequency domain measurements. This method will be introduced in Section 2.2. It works for simple cases; but for systems which work under high frequency and have large number of ports, its performance is poor in terms of stability, CPU cost and memory requirements [67].

2. Recursive convolution method: This approach is divided into two steps: The first step is to generate mathematic macromodel by some fitting techniques. In recent years, numerous algorithms to automatically generate this model are invented and massive work on improving the accuracy and the efficiency of these algorithms had been done. The most appealing ones, among these new algorithms, are vector fitting method (VF) [20] and Loewner matrix method (LM) [43]. Compare to their predecessors, such as least squares method (LS) [12] and asymptotic waveform evaluation method (AWE) [14], these new algorithms are robust enough to handle a complex multi-port system with a large number of ports and with a high bandwidth; they also have many techniques to reduce the CPU cost. Depend on these methods; results can be in format of rational approximation, poles and residues pairs, state matrices and others. These methods will be briefly introduced in Section 2.3 and detailedly mentioned in literature survey part (Chapter 3). When the complicated system would be broke into various less complicated components; and then the second step is to apply recursive convolution on these components for transient analysis. The idea of this method is to simplify the complicate system into numbers of less complicated systems and then apply convolution on them; so this method is much more superior on handling complicated systems comparing to the first approach. However, when the system has a large number of ports and work under high frequency environment, this method would generate huge number of components (systems) and the convolution on each component would be very inefficient.

3. SPICE compatible macromodel is the last the approach, and it is also the main method discussed in this thesis. It obtains mathematic macromodel at first; then in the second step, these results are fitted to the MNA model to generate SPICE netlist. Modified nodal analysis (MNA) equation is a common analytical model for transient analysis and it will be introduced in Section 2.4. After the MNA equation is fitted, SPICE netlist can be obtained through some techniques. So the final simulation result of the unknown system would be list of components of capacitors, resistors, inductors and others. This prototype is very clear and can be easily embedded into other simulation.

However, if the MNA model is directly applied to simulate a complicated system, matrices would have huge-size [6]. It also would generate massive components in the SPICE netlist. An example for this result is given in Section 4.1. To avoid difficulty, we utilize the properties of MNA equation and time-domain macromodels (from VF and LM method) to create optimized methods which can quickly fit macromodels to MNA equation; so the process can be accelerated. These methods are acclaimed in Chapter 4 and 5.

The main purposes of this thesis are to describe these algorithms, to compare their capabilities and to conclude their advantages and disadvantages. As well, the core parts of this thesis are the proposed methods (in Chapter 4 and 5), which can efficiently generate SPICE netlist from time-domain macromodel.

To formulate the problem for the thesis, the left part of Figure 2.2 shows a linear multi-port linear network and simulation method on this network. To solve the network, we can treat it as a black box, as it shows on the right part of Figure 2.2. By looking at the voltages and currents data through their ports, we can 'guess' the internal components of the black box. To achieve this goal, or to establish the analytical time domain macromodel, the first step is to obtain the frequency response data, Y-parameter. These data can be measured through several equipments, such as Network Analyzer, or can be generated through full wave simulation of software, such as HFSS or HFWorks. The data, in frequency domain, can be represented as

$$(s_k, Y(s_k)) \quad k = 1,2,\dots,n \qquad (2.1)$$

Where s_k is the complex frequency and n is the number of frequency points. $Y(s_k) \in \mathbb{C}^{P\times P\times n}$ are the Y-parameter matrices at s_k, P is the number of the ports. To guarantee the availability of the data, s_k must cover the bandwidth of the signal. Increasing the density of frequency points improves the accuracy, but also increases the complexity of the simulation.

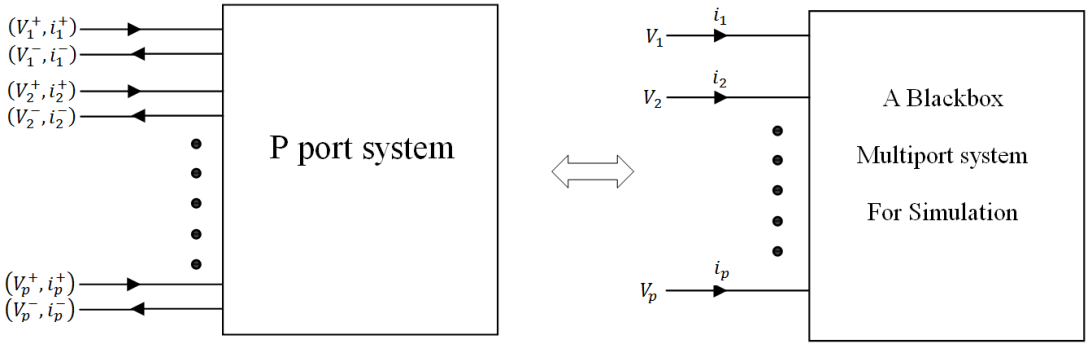


Figure 2.2 Multiport system in frequency domain and Blackbox simulation

The following sections of this chapter introduce background knowledge, based on Figure 2.1, will be introduced in this order: Direct convolution method is introduced in Section 2.2. Various fitting methods are introduced in Section 2.3. It also mentions the state space matrice model and pole-residue approximation model, which are two very important models in the thesis. Their details are given in Appendix I. MNA model will be introduced in Section 2.4. In Section 2.5, linear subsection is discussed to show that the MNA model of this system can be embedded into other systems.

2.2 Direct Convolution Method on Y-parameter

In [34-35, 51], a convolution based approach for the transient analysis of transmission lines described by S-Parameters is presented. In [67], author summarizes these techniques and adapts them on Y-parameter. It uses Inverse Fourier Transform to switch Y-parameter to time domain; Fast Fourier Transform (FFT) or Fast Matrix-Vector Multiplication (FMM) [68, 69] methods can be applied at this step.

Suppose the voltages of all P port of network in Figure 2.2 are $v_1(t), \dots, v_P(t)$, the current of an arbitrary port q, $i_q(t)$, can be described by the definition of Y-parameter, in frequency domain [11]:

$$I_q(s) = \sum_{\varphi=1}^{n} Y_{\varphi q}(s) V_\varphi(s) \qquad (2.2)$$

Apply Fourier transform on this equation, and solve it by FFT or FMM:

$$i_q(t) = F^{-1}\left[\sum_{\varphi=1}^{p} Y_{\varphi q}(s) V_\varphi(s)\right] = \sum_{\varphi=1}^{p} F^{-1}\left[Y_{\varphi q}(s) V_\varphi(s)\right]$$
$$= \sum_{\varphi=1}^{P} \gamma_{g\varphi q}(t) * v_P(t) = \sum_{\varphi=1}^{P} \int_0^t \gamma_{g\varphi q}(t-\tau) v_P(t) d\tau \qquad (2.3)$$

γ_{gmq} is the Green's function of this system and its detail is mentioned in [73]. For the transient analysis, suppose$t_a = a\Delta t$, $t_b = b\Delta t$ and is the step time, then derive (2.3) from t_b to t_a:

$$i_q(t_a) = \sum_{\varphi=1}^{p} \sum_{\tau'=b}^{a} \gamma_{g\varphi q}\left(t_a - \tau'\Delta t\right) v_P\left(\tau'\Delta t\right)\Delta t \qquad (2.4)$$

(2.4) states the result through direct convolution approach; however this approach has two fatal drawbacks for system has large number of ports. First, the convolution and inverse Fourier Transform is very expensive on CPU cost, especially for a long time period simulation; extra memories are also required for storing the time domain signals. With the increase of number of ports, both CPU and memory are rapidly exhausted. The other drawback is that inverse Fourier Transform decreases the accuracy when port number increases, since this algorithm cause aliasing and other problems [67]. For these reasons, direct transient method on multiport system is inefficient for system with a large number of ports; it is also the reason that the current approaches for time-domain analysis is focus on the macromodelling or system identification.

2.3 Various Fitting methods on time-domain simulation

Starting from measured data of frequency points of a multiport system, there are quite a few methods to obtain the time-domain macromodel:

Standard Least Square (LS) Method [12-13] is one of the oldest methods. It assumes the fitting parameter as an approximation of a rational function, and then use iteration to obtain the approximation. The whole process is time consuming and speed to increase accuracy is slow. It is seldom used at present but it provides background knowledge for later method, such as VF method. This method is detailedly introduced in Chapter 3.1.

Strictly positive real (SPR) approximation [32,36-37] also assumes the fitting parameter as an approximation of a rational function, but use Vandermonde matrix to adapt the approximations into state space matrices model. Convex Programming method [38-39] was proposed based on SPR method. It optimizes the state space matrices model by adding more matrices part, so that the accuracy of approximation has been improved. These two methods are introduced in Chapter 3.2.

Asymptotic Waveform Evaluation (AWE) method [14-16] use moment matching techniques to solve the rational approximation function. It expands rational approximation into Maclarurin series and expands moments in Taylor series; then matches the approximation and moments to obtain coefficients of approximation. Since AWE method's results are also in format of rational approximations, we skip this method in this thesis.

Vector Fitting Method is one of most popular fitting method. It derives the rational approximation into numbers of pole-residue groups and fit these coefficients recursively to improve the accuracy of approximation, compare to previous methods, VF method presents high efficiency. This method is introduced in Chapter 3.3.

Loewner Matrix Method applies, at first, tangential interpolations on original data to construct Loewner matrix and shifted Loewner Matrix. Then the state space matrices can be calculated. LM method is very simple and fast. It is also one of most appealing fitting method at present. It is introduced in Chapter 3.4.

Throughout out all these methods, the results various into two formats: State Space Matrices Model and Pole-residue Rational Approximation Model. These two formats are necessary before starting the fitting methods. They are introduced in Appendix I.

2.4 MNA Formulated Circuit Model

MNA formulated circuit model is based on Modified Nodal Analysis (MNA) circuit equation and the formulation of this equation is according to Kirchoff's Current Law (KCL) and Kirchoff's Voltage Law (KVL) at the circuit nodes. The MNA equation for a non-linear system in time domain is:

$$Gx(t) + C\dot{x}(t) + f\big(x(t)\big) = b(t) \quad (2.5)$$

Where

- $x(t) \in \mathbb{R}^m$ is the vector of voltage nodes, voltage source, short circuits, linear inductor currents and other non-linear unknown elements.
- $G, C \in \mathbb{R}^{m\times m}$ are matrices stores the circuit information of linear memoryless and memory elements.
- $f\big(x(t)\big) \in \mathbb{R}^m$ is a vector of scalar function of non-linear components.

$$f\big(x(t)\big) = [f_1(x) \quad f_2(x) \quad \cdots \quad f_m(x)]^T \qquad (2.6)$$

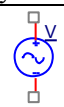
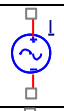
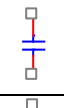
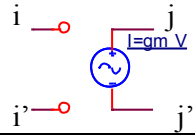
- $b(t) \in \mathbb{R}^m$ is a vector stores independent voltage source and current source.
- m is the total number of unknown variables in this system.

Transfer (2.1) into frequency domain:

$$Gx + sCx + f(x) = b \quad (2.7)$$

MNA equations for linear system doesn't contain component $f\big(x(t)\big)$. To construct (2.7) from an electronic circuit design, all linear components in this design must be written into the matrices through component stamps, which are list in table 2-1. Conversely, equivalent circuit components can be extracted through MNA equations [71-72]. Table 2-1 below has given some important stamps for electronic components.

Table 2-1 MNA Stamps for ideal circuit components [71]

Element	Symbol	Stamp on Matrices in MNA equaiton	Equation
Voltage source		$\boldsymbol{G}$: $\begin{matrix} & col\ i & col\ j & newcol \\ row\ i & & & +1 \\ row\ j & & & -1 \\ new\ row & +1 & -1 & \end{matrix}$ $\boldsymbol{b}$: $\begin{bmatrix} \\ \\ E \end{bmatrix}$	$I = I_i$ $-I = I_j$ $V_i - V_j = E$
Current source		$\boldsymbol{b}$: $\begin{matrix} row\ i \\ row\ j \end{matrix}$ $\begin{bmatrix} -J \\ J \end{bmatrix}$	$J = I_i$ $-J = I_j$
Resistor		$\boldsymbol{G}$: $\begin{matrix} & col\ i & col\ j \\ row\ i & 1/R & -1/R \\ row\ j & -1/R & 1/R \end{matrix}$	$(V_i - V_j)/R = I_i$ $(-V_i + V_j)/R = I_j$
Capacitor		$\boldsymbol{C}$: $\begin{matrix} & col\ i & col\ j \\ row\ i & C & -C \\ row\ j & -C & C \end{matrix}$	$sC(V_i - V_j) = I_i$ $sC(-V_i + V_j) = I_j$
Inductor		$\boldsymbol{C}$: $\begin{matrix} & col\ i & col\ j & newcol \\ row\ i & & & +1 \\ row\ j & & & -1 \\ new\ row & +1 & -1 & -L \end{matrix}$	$I_L = I_i$ $-I_L = I_j$ $V_i - V_j - sLI_L = 0$
Open circuit		No change	
Short circuit		$\boldsymbol{G}$: $\begin{matrix} & col\ i & col\ j & newcol \\ row\ i & & & +1 \\ row\ j & & & -1 \\ new\ row & +1 & -1 & \end{matrix}$	$I = I_i$ $-I = I_j$ $V_i - V_j = 0$
VCCS		$\boldsymbol{G}$: $\begin{matrix} & col\ i & col\ i' \\ row\ j & gm & -gm \\ row\ j' & -gm & gm \end{matrix}$	$gm(V_i - V_{i'}) = I_j$ $gm(-V_i + V_{i'}) = I_{j'}$

The key advantage of MNA Formulated Circuit Model is that the concept of component stamps allows computer automatically switch between MNA equation and equivalent circuit. Other properties are that G and C are sparse matrices and their sizes depend on the number of voltage nodes and source component. To clearly demonstrate the MNA Circuit Model, an example is given below:

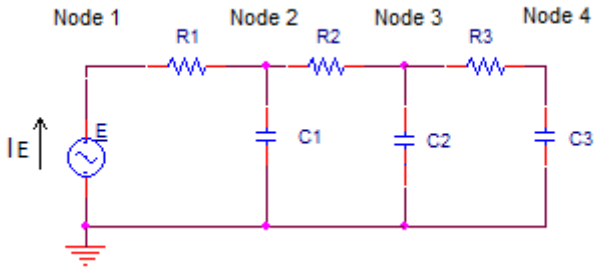


Figure 2.3 Example Circuit for Modified Nodal Analysis

Figure 2.3 gives an example for MNA equation analysis on linear circuit. In this circuit, there are 4 nodes, the voltages on for nodes are V_1~V_4. Applying KCL at each node, we obtain:

$$\frac{V_1-V_2}{R_1} - I_E = 0$$

$$\frac{V_2-V_1}{R_1} + sC_1V_2 + \frac{V_2-V_3}{R_2} = 0$$

$$\frac{V_3-V_2}{R_2} + sC_2V_3 + \frac{V_3-V_4}{R_3} = 0$$

$$\frac{V_4-V_3}{R_3} + sC_3V_4 = 0$$

$$V_1 = E \qquad (2.8)$$

We use conductance g instead of resistance R. Reformat (2.8) into matrices equation, we can obtain MNA equation in matrices format:

$$\begin{bmatrix} g_1 & -g_1 & 0 & 0 & -1 \\ -g_1 & g_1+g_2+sC_1 & -g_2 & 0 & 0 \\ 0 & -g_2 & g_2+g_3+sC_2 & -g_3 & 0 \\ 0 & 0 & -g_3 & g_3+sC_3 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \\ I_E \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ E \end{bmatrix} \quad (2.9)$$

Separate imaginary part, we can obtain:

$$\begin{bmatrix} g_1 & -g_1 & 0 & 0 & -1 \\ -g_1 & g_1+g_2 & -g_2 & 0 & 0 \\ 0 & -g_2 & g_2+g_3 & -g_3 & 0 \\ 0 & 0 & -g_3 & g_3 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \\ I_E \end{bmatrix} + s \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & C_1 & 0 & 0 & 0 \\ 0 & 0 & C_2 & 0 & 0 \\ 0 & 0 & 0 & C_3 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \\ I_E \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ E \end{bmatrix} \quad (2.10)$$

Compare (2.10) to the rules in table 2-1, we observe that components match their values in matrices G and C.

2.5 Linear Subsection

In previous section, we have introduced MNA equations and MNA circuit model (stamps). At dealing complexity of modelled circuit, MNA matrices are sometimes huge. One way to simplify the matrices is to use linear multi-port subsection, which is considered in this section. To illustrate the concept briefly, we have used the example from previous section to observe the effect of linear subsection. If we see the components in the block as a circuit subsection, then a new circuit is generated as below:

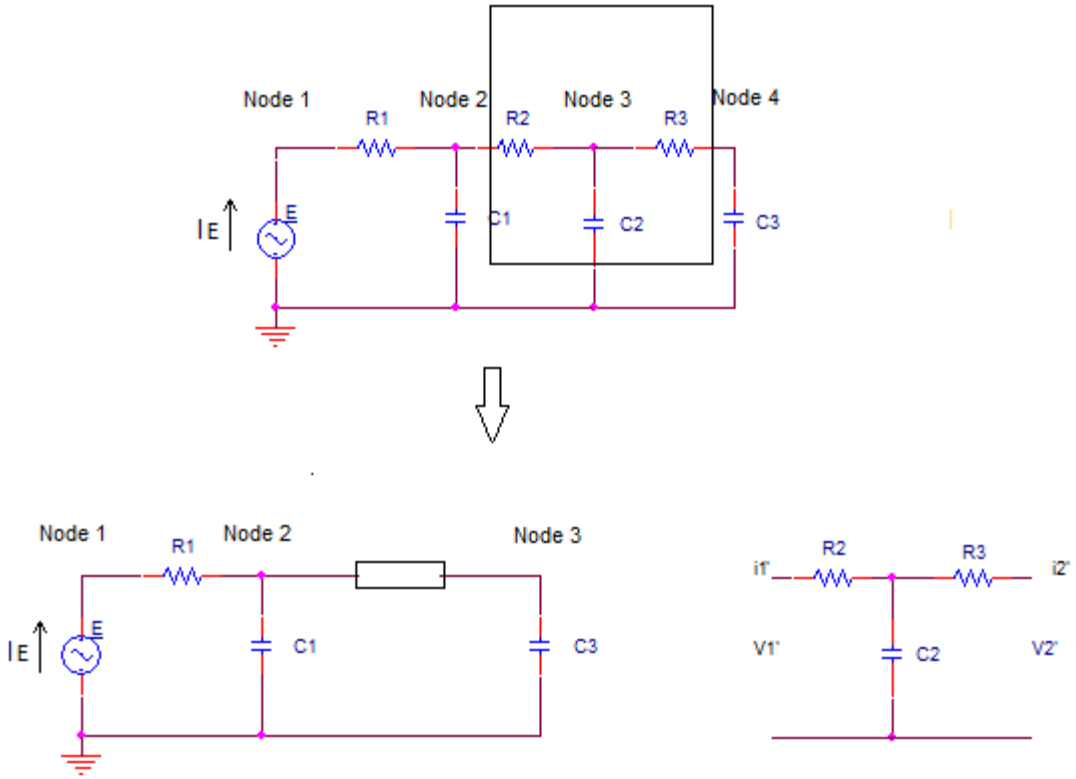


Figure 2.4 Example Circuit for Linear Lump circuit

There are 3 nodes in the new circuit, we rename them to V_1~V_3. On the right side, we draw the internal circuit of the block (lump). At first step, we consider Y-parameter of the lump circuit on the bottom right, it would give:

$$\begin{bmatrix} i_1' \\ i_2' \end{bmatrix} = \begin{bmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{bmatrix} \begin{bmatrix} V_1' \\ V_2' \end{bmatrix} = \begin{bmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{bmatrix} \begin{bmatrix} V_2 \\ V_3 \end{bmatrix}$$

$$y_{11} = \frac{g_2(g_3+sC_2)}{g_2+g_3+sC_2}, y_{12} = y_{21} = \frac{-g_2 g_3}{g_2+g_3+sC_2}, y_{22} = \frac{g_3(g_2+sC_2)}{g_2+g_3+sC_2} \quad (2.11)$$

Insert equations into the MNA equation, we obtain:

$$\begin{bmatrix} g_1 & -g_1 & 0 & 1 \\ -g_1 & g_1 + sC_1 + y_{11} & y_{12} & 0 \\ 0 & y_{21} & sC_3 + y_{22} & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ I_E \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ E \end{bmatrix} \quad (2.12)$$

Compare (2.12) to (2.9), we have observed that the order and complexity of MNA equation has decreased. If the lump is not a simple circuit, but a transmission line or an interconnector; this step would give significant impact on circuit simulation since all Y-parameters become frequency-dependent and have imaginary part in most cases. In this case, macromodelling on the lump is necessary.

Any linear subsection can be completely described in the frequency domain by its network parameters (e.g. Y, Z or S-Parambers). Here we use the Y-parameters to show how a linear subsection can be easily stamped in frequency domain MNA equations.

Generally speaking, this thesis observes the problem in Figure 2.4 reversely. From the frequency measurements of unknown system or subsection, it analyzes different algorithms to predict the inside components for them.

Chapter 3 Literature Survey

As can be seen in Figure 2.1, one of the key steps in the generation of a spice compatible macromodel are the algorithms that start from frequency domain measurements and construct a numerical macromodel by fitting methods. In this Chapter we review several fitting methods of generating time-domain or frequency-domain marcomodels from tabulate Y-parameter data.

3.1 Standard Least Square Method

Standard Least Square (LS) Method obtains the rational approximation model of Y-parameter from its data [12, 13]. In [33], the author has described the formation of pure-real matrices equation through this method. (3.1) is a rational approximation for an element in Y-matrix and N is a guessed order for the rational approximation:

$$Y_{ij}(s) = \frac{N(s)}{D(s)} = \frac{a_0+a_1s+a_2s^2+\cdots+a_Ns^N}{1+b_1s+b_2s^2+\cdots+b_Ns^N};\ a, b \in \mathbb{R}; Y_{ij}(s) \in \mathbb{C} \quad (3.1)$$

The first step is to separate real part and imaginary part of $Y_{ij}(s)$ and $N(s)$:

$$Y_{ij}^{(re)} + jY_{ij}^{(im)} = \frac{N^{(re)}+N^{(im)}}{D} \quad (3.2)$$

$$Y_{ij}^{(re)}D + jY_{ij}^{(im)}D = N^{(re)} + N^{(im)} \quad (3.3)$$

Expand equation (3.3) with a and b parameter:

$$Y_{ij}^{(re)}(1 + b_1s + b_2s^2 + \cdots + b_Ns^N) + jY_{ij}^{(im)}(1 + b_1s + b_2s^2 + \cdots + b_Ns^N)$$
$$= (a_0 + a_2s^2 + \cdots) + (a_1s + a_3s^3 + \cdots) \quad (3.4)$$

Separate the real part and imaginary part on both sides of (3.4):

- N is even

  The real part is:

  $Y_{ij}^{(re)} + jY_{ij}^{(im)}b_1 s + Y_{ij}^{(re)}b_2 s^2 + \cdots + Y_{ij}^{(re)}b_N s^N = a_0 + a_2 s^2 + \cdots + a_N s^N$ (3.5)

  The imaginary part is:

  $jY_{ij}^{(im)} + Y_{ij}^{(re)}b_1 s + jY_{ij}^{(im)}b_2 s^2 + \cdots + jY_{ij}^{(im)}b_N s^N = a_1 s + a_3 s^3 + \cdots + a_{N-1} s^{N-1}$ (3.6)

- N is odd

  The real part is:

  $Y_{ij}^{(re)} + jY_{ij}^{(im)}b_1 s + Y_{ij}^{(re)}b_2 s^2 + \cdots + Y_{ij}^{(re)}b_N s^N = a_0 + a_2 s^2 + \cdots + a_N s^{N-1}$ (3.7)

  The imaginary part is:

  $jY_{ij}^{(im)} + Y_{ij}^{(re)}b_1 s + jY_{ij}^{(im)}b_2 s^2 + \cdots + jY_{ij}^{(im)}b_N s^N = a_1 s + a_3 s^3 + \cdots + a_N s^N$ (3.8)

Next step is to formulate matrix equation $Ax = b$ from equation (3.5) to (3.7). In this step, all the Y-parameter and frequency points must be considered; suppose N is an even number, substitute $s = j\omega$ to (3.5) and (3.6):

$$\left[a_0 - \omega^2 a_2 + \omega^4 a_4 \cdots + (-1)^{\frac{N}{2}}\omega^N a_N\right] - \left[Y_{ij}^{(re)} - Y_{ij}^{(im)}\omega b_1 - Y_{ij}^{(re)}\omega^2 b_2 + \cdots + (-1)^{\frac{N}{2}}Y_{ij}^{(re)}\omega^N b_N\right] = 0 \quad (3.9)$$

$$\left[a_1\omega - \omega^3 a_3 + \omega^5 a_5 \cdots + (-1)^{\left(\frac{N}{2}-1\right)}\omega^{N-1} a_{N-1}\right] - \left[Y_{ij}^{(im)} + Y_{ij}^{(re)}\omega b_1 - Y_{ij}^{(im)}\omega^2 b_2 + \cdots + (-1)^{\frac{N}{2}}Y_{ij}^{(im)}\omega^N b_N\right] = 0 \quad (3.10)$$

(3.9) and (3.10) are elementary equations for construct $Ax = b$, to avoid the Null solution, $Y_{ij}^{(re)}$ and $Y_{ij}^{(im)}$ can be moved to the right side of the matrix.

To avoid floating-points overflow for the problem of Figure 2.3, ω data must first be curved by a normalized frequency ω_{nor} where

$$\omega_{nor} = \sqrt{\omega_{min}\omega_{max}} = \sqrt{\omega_1\omega_n}; \omega_k' = \omega_k/\omega_{nor} \quad (3.11)$$

Then frequency points and Y-data for constructing matrices equation are:

$$\left(\omega_k', Y_{ij}(\omega_k')\right) \quad k = 1,2,\dots,n \quad (3.12)$$

Then the matrices equation $Ax = b$ are:

$A =$

$$\begin{bmatrix}
1 & -\omega_1^2 & \cdots & (-1)^{\frac{N}{2}}\omega_1^N & 0 & 0 & \cdots & 0 & \omega_1 Y_{ij}^{(im)}(j\omega_1) & \omega_1^2 Y_{ij}^{(re)}(j\omega_1) & \cdots & -(-1)^{\frac{N}{2}}\omega_1^N Y_{ij}^{(re)}(j\omega_1) \\
1 & -\omega_2^2 & \cdots & (-1)^{\frac{N}{2}}\omega_2^N & 0 & 0 & \cdots & 0 & \omega_2 Y_{ij}^{(im)}(j\omega_2) & \omega_2^2 Y_{ij}^{(re)}(j\omega_2) & \cdots & -(-1)^{\frac{N}{2}}\omega_2^N Y_{ij}^{(re)}(j\omega_2) \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
1 & -\omega_n^2 & \cdots & (-1)^{\frac{N}{2}}\omega_n^N & 0 & 0 & \cdots & 0 & \omega_n Y_{ij}^{(im)}(j\omega_n) & \omega_n^2 Y_{ij}^{(re)}(j\omega_n) & \cdots & -(-1)^{\frac{N}{2}}\omega_n^N Y_{ij}^{(re)}(j\omega_n) \\
0 & 0 & \cdots & 0 & \omega_1 & -\omega_1^3 & \cdots & (-1)^{\left(\frac{N}{2}-1\right)}\omega_1^{N-1} & -\omega_1 Y_{ij}^{(re)}(j\omega_1) & \omega_1^2 Y_{ij}^{(im)}(j\omega_1) & \cdots & -(-1)^{\frac{N}{2}}\omega_1^N Y_{ij}^{(im)}(j\omega_1) \\
0 & 0 & \cdots & 0 & \omega_2 & -\omega_2^3 & \cdots & (-1)^{\left(\frac{N}{2}-1\right)}\omega_2^{N-1} & -\omega_2 Y_{ij}^{(re)}(j\omega_2) & \omega_2^2 Y_{ij}^{(im)}(j\omega_2) & \cdots & -(-1)^{\frac{N}{2}}\omega_2^N Y_{ij}^{(im)}(j\omega_2) \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & 0 & \omega_n & -\omega_n^3 & \cdots & (-1)^{\left(\frac{N}{2}-1\right)}\omega_n^{N-1} & -\omega_n Y_{ij}^{(re)}(j\omega_n) & \omega_n^2 Y_{ij}^{(im)}(j\omega_n) & \cdots & -(-1)^{\frac{N}{2}}\omega_n^N Y_{ij}^{(im)}(j\omega_n)
\end{bmatrix}$$

$$x = [a_0 \quad a_2 \quad \cdots \quad a_N \quad a_1 \quad a_3 \quad \cdots \quad a_{N-1} \quad b_1 \quad b_2 \quad \cdots \quad b_N]^T$$

$$b = \left[Y_{ij}^{(re)}(\omega_1) \quad Y_{ij}^{(re)}(\omega_2) \quad \cdots \quad Y_{ij}^{(re)}(\omega_n) \quad Y_{ij}^{(im)}(\omega_1) \quad Y_{ij}^{(im)}(\omega_2) \quad \cdots \quad Y_{ij}^{(im)}(\omega_n)\right]^T \quad (3.13)$$

While ω is the angular frequency, all A and b matrices are real. x contains data of rational approximation and can be solved from $Ax = b$. To avoid ill-condition result, matrices equation (3.13) must be an overdetermined system [68]. A must be overdetermined to satisfy the overdetermined condition:

$$A \in \mathbb{R}^{2n\times(2N-1)} \Longrightarrow 2n \leq 2N-1 \Leftrightarrow N \leq n \quad (3.14)$$

So the guessed order N must be smaller than the number of frequency samples n:

Then the overdetermined equation can be solved by:

$$A^T Ax = A^T b$$

$$x = (A^T A)^{-1} A^T b \quad (3.15)$$

Once de-normalized x is obtain, b parameters are taken to calculate poles of Y_{ij}; unstable poles are removed and leave effective poles $\left(p_1, p_2, \dots, p_{N'}\right)$. With these poles, revised b parameters $\left(b_1, b_2, \dots, b_{N'}\right)$ can be obtained. When true approximation of $D(s)$ is known, a parameters ($N(s)$) can be calculated through a similar process, which:

$$Y_{ij}(s)D(s) = N(s) = a_0 + a_1 s + a_2 s^2 + \cdots + a_{N'} s^{N'} \quad (3.16)$$

Assume N' is even, then

$$Re\{Y_{ij}D(j\omega)\} = a_0 - \omega^2 a_2 + \omega^4 a_4 \cdots + (-1)^{\frac{N'}{2}}\omega^{N'} a_{N'};$$

$$Im\{Y_{ij}D(j\omega)\} = \omega a_1 - \omega^3 a_3 + \omega^5 a_5 \cdots + (-1)^{\left(\frac{N'}{2}-1\right)}\omega^{N'-1} a_{N'-1} \quad (3.17)$$

The corresponding matrices equation for (3.17) is:

$$\begin{bmatrix} 1 & -\omega_1^2 & \cdots & (-1)^{\frac{N'}{2}}\omega_1^{N'} & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & -\omega_n^2 & \cdots & (-1)^{\frac{N'}{2}}\omega_n^{N'} & 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & \omega_1 & -\omega_1^3 & \cdots & (-1)^{\left(\frac{N'}{2}-1\right)}\omega_1^{N'-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \omega_n & -\omega_n^3 & \cdots & (-1)^{\left(\frac{N'}{2}-1\right)}\omega_n^{N'-1} \end{bmatrix} \begin{bmatrix} a_0 \\ a_2 \\ \vdots \\ a_{N'} \\ a_1 \\ a_3 \\ \vdots \\ a_{N'-1} \end{bmatrix} = \begin{bmatrix} Re\{Y_{ij}D(j\omega_1)\} \\ \vdots \\ Re\{Y_{ij}D(j\omega_n)\} \\ Im\{Y_{ij}D(j\omega_1)\} \\ \vdots \\ Im\{Y_{ij}D(j\omega_n)\} \end{bmatrix} \quad (3.18)$$

Since $N' \le N \le 2n$, (3.18) is also an overdetermined matrices equation. When a parameters can be solved through (3.15); the rational approximation in (3.1) is obtained.

From effective poles set$(p_1, p_2, \ldots, p_{N'})$, the pole-residue format of Y_{ij} also can be calculated. Suppose poles set have α real poles and β complex conjugate pole pairs. Rewrite Y_{ij} as:

$$\tilde{Y}_{ij}(s) = c + \sum_{t=1}^{N'} \frac{r_t}{s-p_t} = c + \sum_{t=1}^{\alpha} \frac{r_t}{s-p_t} + \sum_{t=\alpha+1}^{\alpha+\beta} \left(\frac{r_t}{s-p_t} + \frac{r_t^*}{s-p_t^*}\right) \quad (3.19)$$

Then (3.17) becomes:

$$Re\{\tilde{Y}_{ij}(s)\} = c + \sum_{t=1}^{\alpha} Re\left(\frac{1}{s-p_t}\right) r_t +$$
$$\sum_{t=\alpha+1}^{\alpha+\beta} \left[Re(r_t) Re\left(\frac{1}{s-p_t} + \frac{1}{s-p_t^*}\right) + Im(r_t) Im\left(\frac{1}{s-p_t^*} - \frac{1}{s-p_t}\right) \right]$$
$$Im\{\tilde{Y}_{ij}(s)\} = \sum_{t=1}^{\alpha} Im\left(\frac{1}{s-p_t}\right) r_t +$$
$$\sum_{t=\alpha+1}^{\alpha+\beta} \left[Re(r_t) Im\left(\frac{1}{s-p_t} + \frac{1}{s-p_t^*}\right) + Im(r_t) Re\left(\frac{1}{s-p_t^*} - \frac{1}{s-p_t}\right) \right] \quad (3.20)$$

Formulate (3.20) into $Ax = b$ format:

$$\begin{bmatrix} 1 & Re\left(\frac{1}{s_1-p_1}\right) & \cdots & Re\left(\frac{1}{s_1-p_\alpha}\right) & Re\left(\frac{1}{s_1-p_{\alpha+1}} + \frac{1}{s_1-p_{\alpha+1}^*}\right) & Im\left(\frac{1}{s_1-p_{\alpha+1}^*} - \frac{1}{s_1-p_{\alpha+1}}\right) & \cdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots \\ 1 & Re\left(\frac{1}{s_n-p_1}\right) & \cdots & Re\left(\frac{1}{s_n-p_\alpha}\right) & Re\left(\frac{1}{s_n-p_{\alpha+1}} + \frac{1}{s_n-p_{\alpha+1}^*}\right) & Im\left(\frac{1}{s_n-p_{\alpha+1}^*} - \frac{1}{s_n-p_{\alpha+1}}\right) & \cdots \\ 0 & Im\left(\frac{1}{s_1-p_1}\right) & \cdots & Im\left(\frac{1}{s_1-p_\alpha}\right) & Im\left(\frac{1}{s_1-p_{\alpha+1}} + \frac{1}{s_1-p_{\alpha+1}^*}\right) & Re\left(\frac{1}{s_1-p_{\alpha+1}^*} - \frac{1}{s_1-p_{\alpha+1}}\right) & \cdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots \\ 0 & Im\left(\frac{1}{s_n-p_1}\right) & \cdots & Im\left(\frac{1}{s_n-p_\alpha}\right) & Im\left(\frac{1}{s_n-p_{\alpha+1}} + \frac{1}{s_n-p_{\alpha+1}^*}\right) & Re\left(\frac{1}{n-p_{\alpha+1}^*} - \frac{1}{s_n-p_{\alpha+1}}\right) & \cdots \end{bmatrix}$$
$$\begin{bmatrix} c \\ r_1 \\ \vdots \\ r_\alpha \\ Re(r_{\alpha+1}) \\ Im(r_{\alpha+1}) \\ \vdots \\ Re(r_{\alpha+\beta}) \\ Im(r_{\alpha+\beta}) \end{bmatrix} = \begin{bmatrix} Re\{Y_{ij}(s_1)\} \\ \vdots \\ Re\{Y_{ij}(s_n)\} \\ Im\{Y_{ij}(s_1)\} \\ \vdots \\ Im\{Y_{ij}(s_n)\} \end{bmatrix} \quad (3.21)$$

Above introduces LS method to obtain the rational approximation for one port data, the advantages are that this method simply formulates real matrices and different initial guessed order N results only small difference on final order. LS method has two major drawbacks: the first is poorness for handling the noise from measured data; a sudden pulse results incorrect or inefficient fitting process [13]. The other drawback is that LS method solve only one entry of data each time, marcomodeling time for a system with a large number of ports is extremely long. For these reasons, LS method usually is applied for low order system. However this method has expanded the ideas for later fitting methods.

3.2 Strictly positive real approximation and Convex Programming method

This section will introduce two methods to generate state space model from Y-parameter. The first one uses strictly positive real (SPR) approximation to generated stable real models and SPR method is given in [32, 36-37]. The second one, Convex programming method, is based on the first one, but develops the superior models on improving stability and passivity [38-39].

Strictly positive real approximation method gives a general method to extract time-domain model in (2.1) from Y-parameter by using rational approximation. SPR method is based on a modified version of a standard nonlinear optimization algorithm [32]. It uses, at first, Levenberg-Marquardt algorithm or LS method to find rational approximation parameter $\tilde{a}$ and $\tilde{b}$, and then uses orthonormal polynomial basis $P_k(s)$ to provide a numerical approximation.

$$\widetilde{\boldsymbol{R}}(\tilde{a},\tilde{b},s)=\frac{\tilde{Y}_{ij}(s)}{\boldsymbol{U}(s)}=\frac{\sum_{k=1}^{m}\tilde{a}_k P_k(s)}{\sum_{k=1}^{n}\tilde{b}_k P_k(s)};\ \tilde{a}\in\mathbb{R}^m,\tilde{b}\in\mathbb{R}^n;\tilde{Y}_{ij}(s)\in\mathbb{C} \quad (3.22)$$

Where $P_k(s)$ is a polynomial basis of order k satisfies (3.23), and $h_{i,j}$ can be obtained through Arnoldi process [32]:

$$sP_{j-1}(s)=\sum_{i=j-1}^{j+1}h_{i,j}P_{i-1}(s) \quad (3.23)$$

If $\boldsymbol{U}_{0:m}(s)$ represents the first $m+1$ columns of Vandermonde matrix[69]:

$$\boldsymbol{U}_{0:m}(s)=\begin{bmatrix}1 & s_1 & s_1^2 & \cdots & s_1^m\\ 1 & s_2 & s_2^2 & \cdots & s_2^m\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ 1 & s_N & s_N^2 & \cdots & s_N^m\end{bmatrix} \quad (3.24)$$

Rewrite (3.23) to adapt format of state space model:

$$\tilde{Y}_{ij}(s)=\left(\sum_{k=1}^{n}\tilde{b}_k P_k(s)\right)\left(\sum_{k=1}^{m}\tilde{a}_k P_k(s)\right)^{-1}\boldsymbol{U}(s)=\boldsymbol{C}(sI-\boldsymbol{A})^{-1}\boldsymbol{B}+\boldsymbol{D} \quad (3.25)$$

State space matrices for ports are constructed as:

$$\begin{aligned}\boldsymbol{A}&=\boldsymbol{H}_n^T-a_n^{-1}h_{n+1,n}\,\boldsymbol{e}_n[a_0 \quad \cdots \quad a_{n-1}]\\ \boldsymbol{B}&=a_n^{-1}h_{n+1,n}\,\boldsymbol{e}_n\\ \boldsymbol{C}&=[b_0 \quad \cdots \quad b_m \quad 0 \quad \cdots \quad 0]\\ \boldsymbol{D}&=0\end{aligned} \quad (3.26)$$

Where $\boldsymbol{H}_n$ is a $n\times n$ upper Hessenberg matrix [32, 69] and $\boldsymbol{e}_n$ is nth column of identity matrix.

For a multiport system with j inputs and i inputs that $Y(s) \in \mathbb{C}^i$, $\boldsymbol{U}(s) \in \mathbb{C}^j$.(3.25) can be rewritten into

$$Y(s) = (\sum_{k=1}^{n} \boldsymbol{B}_k P_k(s))(\sum_{k=1}^{m} \boldsymbol{A}_k P_k(s))^{-1} \boldsymbol{U}(s) \qquad (3.27)$$

Where $\boldsymbol{A}_k \in \mathbb{R}^{j \times j}$, $\boldsymbol{B}_k \in \mathbb{R}^{i \times i}$ and they are the collection of all $\tilde{a}$ and $\tilde{b}$. In this case, state spaces are aggregated with Kronecker product $\otimes$ [32, 69]:

$$\begin{aligned}
&\boldsymbol{A} = \boldsymbol{H}_n^T \otimes \boldsymbol{I}_j - h_{n+1,n} \left(\boldsymbol{e}_n \otimes \boldsymbol{I}_j\right)[\boldsymbol{A}_n^{-1}\boldsymbol{A}_0 \quad \boldsymbol{A}_n^{-1}\boldsymbol{A}_1 \quad \cdots \quad \boldsymbol{A}_n^{-1}\boldsymbol{A}_{n-1}] \\
&\boldsymbol{B} = h_{n+1,n} \left(\boldsymbol{e}_n \otimes \boldsymbol{I}_j\right)\boldsymbol{A}_n^{-1} \\
&\boldsymbol{C} = [\boldsymbol{B}_0 \quad \cdots \quad \boldsymbol{B}_m \quad 0 \quad \cdots \quad 0] \\
&\boldsymbol{D} = 0
\end{aligned} \qquad (3.28)$$

This method directly extracts stable state space matrices from Y-parameter and provides strictly positive real (SPR) approximations on time-domain model; it also can handle high order case which rational approximation has more than 100 poles [32]. The drawback is that SPR method assumes state space matrix $\boldsymbol{D}$ is 0, and then actual time-domain model becomes:

$$\begin{aligned}
sX &= \boldsymbol{A}X + \boldsymbol{B}u \\
y &= \boldsymbol{C}X
\end{aligned} \qquad (3.29)$$

Here $\boldsymbol{D}$ is absorbed by the approximation $\boldsymbol{C}$, so it requires extra number of poles to provide model's accuracy [32]. After [32], a few paper provided optimization method based on this model. In [36], a convex programming approach was proposed to obtain the non-zero approximation for $\boldsymbol{D}$ matrix and $\boldsymbol{Y}^{\infty}$ matrix.

Convex Programming approach takes the approximations of $\boldsymbol{A}$ and $\boldsymbol{B}$ matrices from (3.28), then use them to calculate $\boldsymbol{C}$, $\boldsymbol{D}$ and $\boldsymbol{Y}^{\infty}$ matrices so that these state space matrices could form a guaranteed passive stable model. Similar to (3.25), the approximation formula for a single element $\tilde{Y}_{i,j}(s)$ can be derived as:

$$\tilde{Y}_{i,j}(s_k) = \boldsymbol{C}_i(s_k\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{B}_j + \boldsymbol{D}_{i,j} + s_k\boldsymbol{Y}_{i,j}^{\infty} \qquad (3.30)$$

Construct $\boldsymbol{J}$ matrix that

$$\boldsymbol{J}(s_k) = \begin{bmatrix} \boldsymbol{B}_j^T(s_k\boldsymbol{I} - \boldsymbol{A}^T)^{-1} & \boldsymbol{e}_j^T & s_k\boldsymbol{e}_j^T \end{bmatrix} \qquad (3.31)$$

And $\boldsymbol{X}_i$ is ith column of

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{C}^T \\ \boldsymbol{D}^T \\ (\boldsymbol{Y}^{\infty})^T \end{bmatrix} \qquad (3.32)$$

Since $\boldsymbol{A}$ and $\boldsymbol{B}$ are already known from last process, $\boldsymbol{J}$ matrix is a known matrix. $\boldsymbol{X}$ contains rest of unknown state space matrices. To calculate $\boldsymbol{X}$, the weighted least-squares error of the approximation is used:

$$\sum_{k=1}^{n} \left\| \tilde{Y}_{i,j}(s_k) - Y_{i,j}(s_k) \right\|_2^2 < t_{i,j} \qquad (3.33)$$

Separate real and imaginary part of $\boldsymbol{J}_{i,j}$ and original data $Y_{i,j}$ and input all frequency data, two new matrices $\boldsymbol{F}$ and $\boldsymbol{G}$ can be constructed:

$$\boldsymbol{F}_{i,j}(k) = \begin{bmatrix} Re(\boldsymbol{J}_{i,j}(s_1)) & \cdots & Re(\boldsymbol{J}_{i,j}(s_n)) \\ Im(\boldsymbol{J}_{i,j}(s_1)) & \cdots & Im(\boldsymbol{J}_{i,j}(s_n)) \end{bmatrix}$$

$$\boldsymbol{G}_{i,j}(k) = \begin{bmatrix} Re(Y_{i,j}(s_1)) & \cdots & Re(Y_{i,j}(s_n)) \\ Im(Y_{i,j}(s_1)) & \cdots & Im(Y_{i,j}(s_n)) \end{bmatrix} \quad (3.34)$$

Then weighted least-squares error function (3.33) transforms into

$$\sum_{k=1}^{n} \left\| \tilde{Y}_{i,j}(s_k) - Y_{i,j}(s_k) \right\|_2^2 < t_{i,j} \Longrightarrow \left\| \boldsymbol{F}_{i,j}\boldsymbol{X}_i - \boldsymbol{G}_{i,j} \right\| < t_{i,j} \quad (3.35)$$

If there is a QR decomposition for $\boldsymbol{F}$ that

$$\boldsymbol{F}_{i,j} = \boldsymbol{Q}_{i,j}\boldsymbol{R}_{i,j} \quad (3.36)$$

Least-squares error function further transforms to

$$\left\| \boldsymbol{F}_{i,j}\boldsymbol{X}_i - \boldsymbol{G}_{i,j} \right\| = \left(\boldsymbol{F}_{i,j}\boldsymbol{X}_i - \boldsymbol{G}_{i,j} \right)^T \left(\boldsymbol{F}_{i,j}\boldsymbol{X}_i - \boldsymbol{G}_{i,j} \right)$$

$$= \left(\boldsymbol{R}_{i,j}\boldsymbol{X}_i - \boldsymbol{Q}_{i,j}^T\boldsymbol{G}_{i,j} \right)^T \left(\boldsymbol{R}_{i,j}\boldsymbol{X}_i - \boldsymbol{Q}_{i,j}^T\boldsymbol{G}_{i,j} \right) + \boldsymbol{G}_{i,j}^T\left(\boldsymbol{I} - \boldsymbol{Q}_{i,j}\boldsymbol{Q}_{i,j}^T \right)\boldsymbol{G}_{i,j}$$

$$= \boldsymbol{E}_{i,j}^T\boldsymbol{E}_{i,j} + \delta_{i,j}^2 \quad (3.37)$$

Finally least constrains (3.35) becomes:

$$\boldsymbol{E}_{i,j}^T\boldsymbol{E}_{i,j} + \delta_{i,j}^2 < t_{i,j} \quad (3.38)$$

Where

$$\boldsymbol{E}_{i,j} = \boldsymbol{R}_{i,j}\boldsymbol{X}_i - \boldsymbol{Q}_{i,j}^T\boldsymbol{G}_{i,j};\ \delta_{i,j}^2 = \boldsymbol{G}_{i,j}^T\left(\boldsymbol{I} - \boldsymbol{Q}_{i,j}\boldsymbol{Q}_{i,j}^T \right)\boldsymbol{G}_{i,j} \quad (3.39)$$

To solve $\boldsymbol{C}, \boldsymbol{D}$ and $\boldsymbol{Y}^{\infty}$ matrices, the author has used Structure-Exploiting Formulation (SEF) method, which has proposed in [36,39]. Compare to SPR method, convex programming method provides better approximation for $\boldsymbol{C}$ and adds approximations for $\boldsymbol{D}$ and $\boldsymbol{Y}^{\infty}$; which improves model's accuracy. Furthermore, it provides a guaranteed passive stable approximation. From the simulation, it provides extraordinary result for single-input single-output system; however, this method only handle small problem. For system with a large number of ports ($N \geq 40$), accumulation of $t_{i,j}$ would cause inefficiency and inaccuracy of the approximation [39].

3.3 Vector Fitting Method

Vector Fitting (VF) method is developed by Bjǿrn Gustavsen and Adam Semlyen in 1996 [20-24]. VF algorithm is based on Sanathanan-Keorner method [22]; it estimates and iteratively refines poles and residues until desired accuracy is achieved. It is also capable to handle large system (over 100ports). After original algorithm is proposed in [20], several new modified VF algorithms are invented that gigantically accelerates the speed and improve the capacity [25-31].

3.3.1 Algorithm implementation

As it mentioned in [20-24], VF method generates poles and residues for rational approximation, where is

$$f(s) \approx \frac{a_0+a_1 s+a_2 s^2+\cdots+a_N s^N}{b_0+b_1 s+b_2 s^2+\cdots+b_N s^N} \qquad (3.40)$$

With measure data

$$\left(s_{fp}, F(s_{fp})\right), fp = 1,2,\ldots,k \qquad (3.41)$$

If (3.61) is written into pole-residue format:

$$f(s) = \sum_{n=1}^{N} \frac{r_n}{s-p_n} + d + sh \qquad (3.42)$$

Then VF method defines a weight function:

$$\sigma(s) = \sum_{n=1}^{N} \frac{\tilde{r}_n}{s-\tilde{p}_n} + 1 \quad (3.43)$$

Weight function σ is from our guess; in other words, $\bar{r}_n$ and $\bar{p}_n$ are known parameters. If $f(s)$ is introduced to $\sigma(s)$:

$$\sigma_{fit}(s) f(s) \cong \sigma f_{fit}(s) = \sum_{n=1}^{N} \frac{r_n}{s-\tilde{p}_n} + d + sh \quad (3.44)$$

Thus

$$f(s) \cong \frac{\sigma f_{fit}(s)}{\sigma(s)} \qquad (3.45)$$

If all the parameter in (3.66) are written into pole-zero format [20]:

$$f(s) = h\frac{\prod_{n=1}^{N+1}(s-z_n)}{\prod_{n=1}^{N}(s-p_n)}; \sigma f_{fit}(s) = h\frac{\prod_{n=1}^{N+1}(s-z_n)}{\prod_{n=1}^{N}(s-\tilde{p}_n)}; \sigma(s) = \frac{\prod_{n=1}^{N}(s-\tilde{z}_n)}{\prod_{n=1}^{N}(s-\tilde{p}_n)} \qquad (3.46)$$

Substitute all equations in (3.46) to (3.45):

$$f(s) \cong \frac{\sigma f_{fit}(s)}{\sigma(s)} \Leftrightarrow h\frac{\prod_{n=1}^{N+1}(s-z_n)}{\prod_{n=1}^{N}(s-p_n)} \cong h\frac{\prod_{n=1}^{N+1}(s-z_n)}{\prod_{n=1}^{N}(s-\tilde{z}_n)} \Longrightarrow p_n \cong \tilde{z}_n \qquad (3.47)$$

To explain the transformation in (3.46) and (3.47):

- $\sigma f_{fit}(s)$ has unknown zeros of $f(s)$ as numerator and known poles of $\sigma(s)$ as denominator;
- $\sigma(s)$ has known zeros as numerator and known poles as denominator.

When divided $\sigma f_{fit}(s)$ by $\sigma(s)$, the known poles of $\sigma(s)$ cancel out and leave $f(s)$ equals to unknown zeros of $f(s)$ as numerator and known zeros of $\sigma(s)$ as denominator. So unknown poles of $f(s)$ is found.

Therefore, VF method can be divided into 4 steps. The flowchart for the process is given in Figure 3.1. The description for each step is followed after the flowchart.

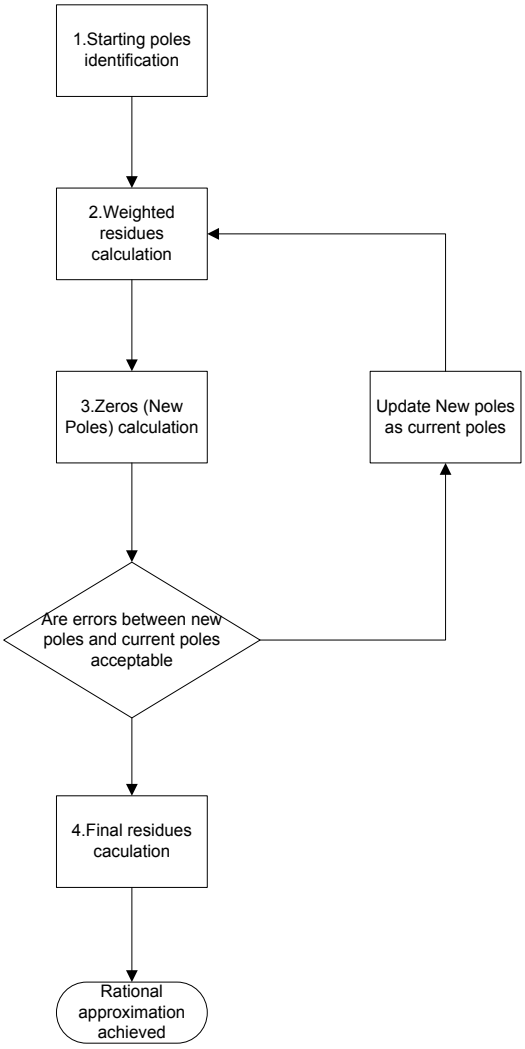


Figure 3.1 Flowchart of Vector fitting method

1) Starting poles identification

At first, number of the guessed poles N must be determined, larger N increase the accuracy of the result but highly increase the amount of calculation, or computation time. Constraints of N will be discussed later in this chapter.

The author of vector fitting algorithm gives some notices for starting poles:

- If the all starting poles are real, the linear problem Ax = b during the calculation will be ill-conditioned and will result inaccurate solution [20].
- A large difference between the starting poles and correct poles may result a poor fitting [20]. It will increase the iteration times to obtain the satisfied result.

The authors also recommend a solution for determining the starting poles. The starting poles are all complex conjugate pairs that [20]:

$$p_n = -\alpha + j\beta_{np};\ p_{n+1} = -\alpha - j\beta_{np} \quad (3.48)$$

Where

$$\alpha = \beta_{np}/100 \quad (3.49)$$

And β is linearly distributed over the frequency range s_{fp} that

$$\beta_{np} = \left[s_1 + \left(\frac{s_k - s_1}{N/2}\right) np\right]/2\pi \text{ , } N \text{ is even} \quad (3.50)$$

2) Weighted residues calculation

Expand (3.44) and reformat the formula:

$$f(s) \cong \sum_{n=1}^{N} \frac{r_n}{s-\tilde{p}_n} + d + sh - f(s)\sum_{n=1}^{N} \frac{\tilde{r}_n}{s-\tilde{p}_n} \quad (3.51)$$

Write (3.51) into into $Ax = b$ format:

$$A = \left[\frac{1}{s-\tilde{p}_1} \dots \frac{1}{s-\tilde{p}_N} \quad 1\ s\ \frac{-f(s)}{s-\tilde{p}_1} \dots \frac{-f(s)}{s-\tilde{p}_N}\right], x = [\,r_1 \dots r_N\ d\ h\ \tilde{r}_1 \dots \tilde{r}_N]^T, b = f(s) \quad (3.52)$$

Apply measured data in (3.41) into matrices equation:

$$\begin{bmatrix} \frac{1}{s_1-\tilde{p}_1} & \cdots & \frac{1}{s_1-\tilde{p}_N} & 1 & s_1 & \frac{-F(s_1)}{s_1-\tilde{p}_1} & \cdots & \frac{-F(s_1)}{s_1-\tilde{p}_N} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{1}{s_k-\tilde{p}_1} & \cdots & \frac{1}{s_k-\tilde{p}_N} & 1 & s_k & \frac{-F(s_k)}{s_k-\tilde{p}_1} & \cdots & \frac{-F(s_k)}{s_k-\tilde{p}_N} \end{bmatrix} \begin{bmatrix} r_1 \\ \vdots \\ r_N \\ d \\ h \\ \tilde{r}_1 \\ \vdots \\ \tilde{r}_N \end{bmatrix} = \begin{bmatrix} F(s_1) \\ \vdots \\ F(s_k) \end{bmatrix} \quad (3.53)$$

The size of matrix A is $A \in \mathbb{C}^{k\times(2N+2)}$, N must be satisfy

$$2N + 2 < k \quad (3.54)$$

So (3.74) becomes an overderterminated equation and x will not be ill-conditioned.

Solving overderterminated equation by

$$(A^T A)x = (A^T b) \quad (3.55)$$

$[\tilde{r}_1 \ldots \tilde{r}_N]$ can be obtained for the next step.

For a complex system, solving (3.53) could be very complicated. A simple method is proposed to transform it into a real matrices equation. Since conjugate complex poles-residues pairs are used, (3.52) can be written as:

$$\bar{A} = \left[\left(\frac{1}{s-\tilde{p}_1}+\frac{1}{s-\tilde{p}_1^*}\right)\left(\frac{j}{s-\tilde{p}_1}-\frac{j}{s-\tilde{p}_1^*}\right)\ldots\left(\frac{1}{s-\tilde{p}_N}+\frac{1}{s-\tilde{p}_N^*}\right)\left(\frac{j}{s-\tilde{p}_N}-\frac{j}{s-\tilde{p}_N^*}\right) 1 \; s_k \left(\frac{-f(s)}{s-\tilde{p}_1}+\frac{-f(s)}{s-\tilde{p}_1^*}\right)\left(\frac{-f(s)j}{s-\tilde{p}_1}-\frac{-f(s)j}{s-\tilde{p}_1^*}\right)\ldots\left(\frac{-f(s)}{s-\tilde{p}_N}+\frac{-f(s)}{s-\tilde{p}_N^*}\right)\left(\frac{-f(s)j}{s-\tilde{p}_N}-\frac{-f(s)j}{s-\tilde{p}_N^*}\right)\right]$$

$$\bar{x} = [Re(r_1)\ Im(r_1) \ldots\ Re(r_N)\ Im(r_N)\ d\ h\ Re(\tilde{r}_1)\ Im(\tilde{r}_1)\ \ldots\ Re(\tilde{r}_N)\ Im(\tilde{r}_N)]^T$$

$$b = f(s) \quad (3.56)$$

Now $\bar{x}$ becomes a real column, separate real part and imaginary part of A and b so that a real matrices equation is obtained:

$$\hat{A} = \begin{bmatrix} Re(\bar{A}) \\ Im(\bar{A}) \end{bmatrix}, \hat{b} = \begin{bmatrix} Re(b) \\ Im(b) \end{bmatrix}, \hat{A}\bar{x} = \hat{b} \quad (3.57)$$

Where $\hat{A} \in \mathbb{R}^{2k\times(2N+2)}, x \in \mathbb{R}^{2N+2}, \hat{b} \in \mathbb{R}^{2k}$

3) Zero calculation:

Calculate zeros of weight function $\tilde{z}_n$ in (3.57), the definition of eigenvalue are used.

$$\sigma(s) = \sum_{n=1}^{N} \frac{\tilde{r}_n}{s-\tilde{p}_n} + 1 = \frac{\prod_{n=1}^{N}(s-\tilde{z}_n)}{\prod_{n=1}^{N}(s-\tilde{p}_n)} \quad (3.58)$$

[20] has given this process as

$$\tilde{z} = eig(A - BC) \quad (3.59)$$

Where $A = \begin{bmatrix} A & 0 \\ 0 & A_p \end{bmatrix}, B = \begin{bmatrix} B \\ B_p \end{bmatrix}, C = [C \quad C_p]$

If $\tilde{p}_n, \tilde{r}_n$ are real, $A_p = \tilde{p}_n, B_p = 1, C_p = \tilde{r}_n$.

If $\tilde{p}_n, \tilde{r}_n$ are complex, or $\tilde{p}_n$ and $\tilde{p}_{n+1}$ are complex conjugate pairs that $\tilde{p}_n = \alpha_i + j\beta_i, \tilde{p}_{n+1} = \alpha_i - j\beta_i$; then

$A_p = \begin{bmatrix} Re(\tilde{p}_n) & Im(\tilde{p}_n) \\ -Im(\tilde{p}_n) & Re(\tilde{p}_n) \end{bmatrix} = \begin{bmatrix} \alpha_i & \beta_i \\ -\beta_i & \alpha_i \end{bmatrix}, B_p = \begin{bmatrix} 2 \\ 0 \end{bmatrix}, C_p = [Re(\tilde{r}_n)\ Im(\tilde{r}_n)]$.

In simulation, sometimes unstable poles are generated in this step, to guarantee the process is stable. The sign of unstable poles are switched to be stable:

$$p_n = \begin{cases} \tilde{z}_n & if\ Re(\tilde{z}_n) \le 0 \\ -Re(\tilde{z}_n) + Im(\tilde{z}_n) & if\ Re(\tilde{z}_n) > 0 \end{cases} \quad (3.60)$$

The rms error between old poles $\tilde{p}_n$ and new poles p_n is

$$\varepsilon_p = \sqrt{\frac{1}{N}\sum_{n=1}^{N}\left|p_n - \tilde{p}_n\right|^2} \quad (3.61)$$

4) Final residues calculation

When satisfied results of p_n are obtained above, the last step is to calculate true value of r_n. Similarly, $Ax = b$ matrices equation is constructed as:

$$\begin{bmatrix} \frac{1}{s_1-p_1} & \cdots & \frac{1}{s_1-p_N} & 1 & s_1 \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ \frac{1}{s_k-p_1} & \cdots & \frac{1}{s_k-p_N} & 1 & s_k \end{bmatrix} \begin{bmatrix} r_1 \\ \vdots \\ r_N \\ d \\ h \end{bmatrix} = \begin{bmatrix} F(s_1) \\ \vdots \\ F(s_k) \end{bmatrix} \quad (3.62)$$

It also can be transformed into real matrices equation by following the method in (3.56-37).

Above parts explain the full algorithm for VF method, the rms error of this approximation is given as [21]:

$$\varepsilon_r = \sqrt{\frac{1}{k}\sum_{np=1}^{k} \left| \left(\sum_{n=1}^{N} \frac{r_n}{s_{np}-p_n} + d + s_{np}h \right) - F(s_{np}) \right|^2} \quad (3.63)$$

Although VF is a robust method to obtain accurate approximation, its drawbacks include convergence and sparse results.

From (3.59), new poles are calculated from $\tilde{z} = eig(A - BC)$. However, this function would return the eigen values in an ascending order, which may change the original order of residue sequence [20]. If the problem is shown graphically:

$$\tilde{p}_i \rightarrow r_i \rightarrow z_? \rightarrow p_?$$

For this reason, convergence can't be determined since a guessed pole is not guaranteed to generate the same pole during the process. In [21], the authors suggested two conditions to stop the iteration:

1) Set the iteration time a significant number.(Normally 20)
2) Compare the difference of old poles and new poles, if the difference (RMS error) is small, stop the iteration.

Another important drawback of VF method is that the state space matrices generated from VF method are very sparse and with huge size. Using the example in 3.74, state space matrix $A \in \mathbb{R}^{5000\times5000}$ is very sparse that all values are at diagonal block. These sparse matrices occur because of overestimation properties of VF method.

3.3.2 VF method on Y-parameter of multiport system

Back to formulated problem in Section 2.3, VF method is applied on P port system. This section has reviewed VF method on Y-parameter of a multiport system. If Y-parameter of a P port system is given in

$$Y(s) = \begin{bmatrix} Y_{11}(s) & Y_{12}(s) & \cdots & Y_{1P}(s) \\ Y_{21}(s) & Y_{22}(s) & \cdots & Y_{2P}(s) \\ \vdots & \vdots & \ddots & \vdots \\ Y_{P1}(s) & Y_{P2}(s) & \cdots & Y_{PP}(s) \end{bmatrix} \quad (3.64)$$

The rational approximation of any Y-parameter is:

$$Y_{AB}(s) = \sum_{n=1}^{N} \frac{r_n^{(AB)}}{s-p_n} + d^{(AB)} + sh^{(AB)}; A, B \in [1, P] \quad (3.65)$$

By following four steps in the flowchart, the method of starting poles and weight function don't change. With the measured data in (2.1); $Ax = b$ matrices equation can be constructed as:

$$A = \begin{bmatrix} \frac{1}{s_1-\tilde{p}_1} & \cdots & \frac{1}{s_1-\tilde{p}_N} & 1 & s_1 & 0 & \cdots & 0 & 0 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \frac{-Y_{11}(s_1)}{s_1-\tilde{p}_1} & \cdots & \frac{-Y_{11}(s_1)}{s_1-\tilde{p}_N} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots & \ddots & \vdots \\ \frac{1}{s_k-\tilde{p}_1} & \cdots & \frac{1}{s_k-\tilde{p}_N} & 1 & s_1 & 0 & \cdots & 0 & 0 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \frac{-Y_{11}(s_k)}{s_k-\tilde{p}_1} & \cdots & \frac{-Y_{11}(s_k)}{s_k-\tilde{p}_N} \\ 0 & \cdots & 0 & 0 & 0 & \frac{1}{s_1-\tilde{p}_1} & \cdots & \frac{1}{s_1-\tilde{p}_N} & 1 & s_1 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \frac{-Y_{12}(s_1)}{s_1-\tilde{p}_1} & \cdots & \frac{-Y_{12}(s_1)}{s_1-\tilde{p}_N} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & 0 & \frac{1}{s_k-\tilde{p}_1} & \cdots & \frac{1}{s_k-\tilde{p}_N} & 1 & s_1 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \frac{-Y_{12}(s_k)}{s_k-\tilde{p}_1} & \cdots & \frac{-Y_{12}(s_k)}{s_k-\tilde{p}_N} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & \frac{1}{s_1-\tilde{p}_N} & \cdots & \frac{1}{s_1-\tilde{p}_N} & 1 & s_1 & \frac{-Y_{PP}(s_1)}{s_1-\tilde{p}_1} & \cdots & \frac{-Y_{PP}(s_1)}{s_1-\tilde{p}_N} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & \frac{1}{s_k-\tilde{p}_N} & \cdots & \frac{1}{s_k-\tilde{p}_N} & 1 & s_1 & \frac{-Y_{PP}(s_k)}{s_k-\tilde{p}_1} & \cdots & \frac{-Y_{PP}(s_k)}{s_k-\tilde{p}_N} \end{bmatrix}$$

$$x = \begin{bmatrix} r_1^{(11)} & \cdots & r_N^{(11)} & d^{(11)} & h^{(11)} & r_1^{(12)} & \cdots & r_N^{(12)} & d^{(12)} & h^{(12)} & \cdots & r_1^{(PP)} & \cdots & r_N^{(PP)} & d^{(PP)} & h^{(PP)} & \tilde{r}_1 & \cdots & \tilde{r}_N \end{bmatrix}^T$$

$$b = [Y_{11}(s_1) \quad Y_{11}(s_2) \quad \cdots \quad Y_{11}(s_k) \quad Y_{12}(s_1) \quad Y_{12}(s_2) \quad \cdots \quad Y_{12}(s_k) \quad \cdots \quad Y_{PP}(s_1) \quad Y_{PP}(s_2) \quad \cdots \quad Y_{PP}(s_k)]^T$$

(3.66)

Similarly, real matrices equation $\hat{A}\bar{x} = \hat{b}$ can be derived from (3.66). Same process is followed that $\tilde{r}_n$ is obtained to calculate zeros until satisfied p_n are obtained. To determine the final residues, equation (3.62) will be repeated for P^2 times for all the parameters in Y-matrix. In most case, Y-matrix is symmetric, so P^2 times can be reduced to$\left(\frac{P^2+P}{2}\right)$ if upper triangle matrix is considered.

$$\begin{bmatrix} \frac{1}{s_1-p_1} & \cdots & \frac{1}{s_1-p_N} & 1 & s_1 \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ \frac{1}{s_k-p_1} & \cdots & \frac{1}{s_k-p_N} & 1 & s_k \end{bmatrix} \begin{bmatrix} r_1^{(AB)} \\ \vdots \\ r_N^{(AB)} \\ d^{(AB)} \\ h^{(AB)} \end{bmatrix} = \begin{bmatrix} Y_{AB}(s_1) \\ \vdots \\ Y_{AB}(s_k) \end{bmatrix}, A \le B,$$

$$Y_{AB}(s) = Y_{BA}(s) \quad (3.67)$$

3.4 Loewner Matrix Method

Loewner Matrix (LM) method is a data fitting method which uses tangential interpolation under the Loewner matrix pencil framework [43-46]. LM method is proposed by Sanda Lefteriu and Athanasios C. Antoulas in 2008. As a very recent algorithm, the goal of LM method is to handle high-complexity system. Compare to VF, who overestimates the problem to achieve the accuracy, LM construct models with less complexity with selectable level of accuracy. Another important property of LM is that LM method directly generates state space matrices for time-domain model. Although LM method only has been proposed from few years, it becomes one of most appealing algorithm in macromodelling.

3.5.1 Algorithm implementation

While most of present algorithms are using least-square approximations; Loewner matrix method uses tangential interpolation and the Loewner matrix pencil.
At First, tangential interpolation is applied on original data to construct Loewner matrix and shifted Loewner Matrix [43].
P port system with data $(s_k, Y(s_k))$ $k = 1,2,\dots,n$ are separated into left and right tangential interpolation. Right tangential interpolation is:

$$\{(\lambda_i, r_i, w_i)|\lambda_i \in \mathbb{C}, r_i \in \mathbb{C}^{m\times 1}, w_i \in \mathbb{C}^{P\times 1}, i = 1,\cdots,l\}$$

$$\Lambda = diag[\lambda_1,\cdots,\lambda_l] \in \mathbb{C}^{l\times l}, R = [r_1,\cdots,r_l] \in \mathbb{C}^{m\times l}, W = [w_1,\cdots,w_l] \in \mathbb{C}^{P\times l} \quad (3.68)$$

And left tangential interpolation is:

$$\{(\mu_j, \ell_j, v_j)|\mu_j \in \mathbb{C}, \ell_j \in \mathbb{C}^{1\times P}, v_j \in \mathbb{C}^{1\times m}, j = 1,\cdots,h\}$$

$$M = diag[\mu_1,\cdots,\mu_h] \in \mathbb{C}^{h\times h}, L = \left[\ell_1^T,\cdots,\ell_h^T\right]^T \in \mathbb{C}^{h\times P}, V = \left[v_1^T,\cdots,v_h^T\right]^T \in \mathbb{C}^{h\times m} \quad (3.69)$$

In these interpolations, $\lambda_i, \mu_j \in s_k$ are frequency points and

$$s_k = \{\lambda_1,\cdots,\lambda_l\} \cup \{\mu_1,\cdots,\mu_h\}, l + h = k \quad (3.70)$$

r_i, ℓ_j are tangential direction vectors; w_i and v_j are tangential vector data and they define the constrains:

$$H(\lambda_i)r_i = [C(\lambda_i E - A)^{-1}B + D]r_i = w_i$$

$$\ell_j H(\mu_j) = \ell_j\left[C(\mu_j E - A)^{-1}B + D\right] = v_i \quad (3.71)$$

Author suggests a selection that

$$\{(\lambda_i = j\omega_i = s_i, r_i, w_i = H(\lambda_i)r_i)|\lambda_i \in \mathbb{C}, r_i \in \mathbb{C}^{m\times 1}, w_i \in \mathbb{C}^{P\times 1}, i = 1,\cdots,n\},$$

$$\left\{(\mu_i = -\lambda_i, \ell_i = r_i^*, v_i = r_i^*\bar{H}(\lambda_i))|\mu_i \in \mathbb{C}, \ell_i \in \mathbb{C}^{1\times P}, v_i \in \mathbb{C}^{1\times m}, j = 1,\cdots,n\right\} \quad (3.72)$$

The Loewner matrix $\mathbb{L}$ and Shifted Loewner matrix $\sigma\mathbb{L}$ can be constructed by:

$$\mathbb{L} = \begin{bmatrix} \frac{v_1 r_1 - \ell_1 w_1}{\mu_1 - \lambda_1} & \cdots & \frac{v_1 r_l - \ell_1 w_l}{\mu_1 - \lambda_l} \\ \vdots & \ddots & \vdots \\ \frac{v_h r_1 - \ell_h w_1}{\mu_h - \lambda_1} & \cdots & \frac{v_h r_l - \ell_h w_l}{\mu_h - \lambda_l} \end{bmatrix} \text{and } \sigma\mathbb{L} = \begin{bmatrix} \frac{\mu_1 v_1 r_1 - \lambda_1 \ell_1 w_1}{\mu_1 - \lambda_1} & \cdots & \frac{\mu_1 v_1 r_l - \lambda_l \ell_1 w_l}{\mu_1 - \lambda_l} \\ \vdots & \ddots & \vdots \\ \frac{\mu_h v_h r_1 - \lambda_1 \ell_h w_1}{\mu_h - \lambda_1} & \cdots & \frac{\mu_h v_h r_l - \lambda_l \ell_h w_l}{\mu_h - \lambda_l} \end{bmatrix} \quad (3.73)$$

They satisfy two Sylvester equations

$$\mathbb{L}\Lambda - M\mathbb{L} = LW - VR, \text{ and}$$

$$\sigma\mathbb{L}\Lambda - M\sigma\mathbb{L} = LW\Lambda - MVR \quad (3.74)$$

Assume $l = h$ and $det(x\mathbb{L} - \sigma\mathbb{L}) \neq 0$, $x \in s_k$, then state space matrices equal to $E = -\mathbb{L}, A = -\sigma\mathbb{L}, B = V, C = W$ and $D = 0$.

In [43], authors gave the solution of the rational interpolation problem and recursive interpolation in the Loewner framework to obtain results.

Later, they give detailed algorithm for guaranteed real implementation in [44] that

$$\left\{\hat{\lambda} = j\omega_i, -j\omega_i; \hat{r} = r_i, r_i; \widehat{w} = H(\lambda_i)r_i, \overline{H}(\lambda_i)r_i; i = 1, \cdots, n/2\right\} \quad (3.75)$$

n is even.Correspondingly,

$$\Lambda = diag\left[j\omega_1, -j\omega_1, \cdots, j\omega_{n/2}, -j\omega_{n/2}\right] \in \mathbb{C}^{n\times n},$$

$$R = \left[r_1, r_1, \cdots, r_{n/2}, r_{n/2}\right] \in \mathbb{C}^{P\times n},$$

$$W = \left[H(\lambda_1)r_1, \overline{H}(\lambda_1)r_1 \cdots, H\left(\lambda_{n/2}\right)r_{n/2}, \overline{H}\left(\lambda_{n/2}\right)r_{n/2}\right] \in \mathbb{C}^{P\times n} \quad (3.76)$$

Left interpolations are

$$\left\{\hat{\mu} = j\omega_{i+n/2}, -j\omega_{i+n/2}; \hat{\ell} = \ell_i, \ell_i; \hat{v} = \ell_i H(\mu_i), \ell_i \overline{H}(\mu_i); i = 1, \cdots, n/2\right\} \quad (3.77)$$

Construct transformation matrix $\widehat{\Pi}$:

$$\widehat{\Pi} = diag[\Pi, \ldots, \Pi] \in \mathbb{C}^{n\times n}, \Pi = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -j \\ 1 & j \end{bmatrix}, \Pi^{-1} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ j & -j \end{bmatrix} \quad (3.78)$$

Apply these transformations to obtain all real Loewner matrix pencils:

$$\hat{\Lambda} = \widehat{\Pi}^{-1}\Lambda\widehat{\Pi}, \hat{R} = R\widehat{\Pi}, \widehat{W} = W\widehat{\Pi}, \widehat{M} = \widehat{\Pi}^{-1}M\widehat{\Pi},$$

$$\hat{L} = L\widehat{\Pi}, \hat{V} = V\widehat{\Pi}, \widehat{\mathbb{L}} = \widehat{\Pi}^{-1}\mathbb{L}\widehat{\Pi}, \text{ and } \sigma\widehat{\mathbb{L}} = \widehat{\Pi}^{-1}\sigma\widehat{\mathbb{L}}\widehat{\Pi} \quad (3.79)$$

The next step is to extract the regular part of Loewner matrices since pencil becomes singular while the measurements are too much. This step also reduces the order of all state space matrices, consider obtained from singular value decomposing (SVD) of

$$[x\mathbb{L} - \sigma\mathbb{L}] = Y_s\Sigma_s X_s \cong Y_1\Sigma_1 X_1, x \in \{\mu_i\} \cup \{\lambda_i\} \quad (3.80)$$

The rank rk of Σ, defines the dimension of regular part of $[x\mathbb{L} - \sigma\mathbb{L}]$

The realized state space matrices are:

$$E = -Y_1^T\mathbb{L}X_1, A = -Y_1^T\sigma\mathbb{L}X_1, B = Y_1^T V, C = WX_1 \text{ and } D = 0 \quad (3.81)$$

Compare to VF method, which also treats system as a black box; Loewner matrix method doesn't require a good estimation of starting poles or any extra input data. The state space matrices have stable size which depends on the input data, while the sizes of those generated from VF method depend on number of guessed poles. The drawback of LM method is that this model requires original data cover full bandwidth. For some systems like transmission lines or distributed systems, low frequency data are hard to achieve and are, sometimes, incorrect. In the next section, a modified algorithm is introduced to solve distributed networks.

3.5.2 Modified Loewner Matrix Method on multiport system

As it mention above, LM method, proposed in [47], provide modified LM method for distributed transmission line networks. From method in previous section, D is zero. At lower bandwidth system, D is embedded in E, A, C so that system generates very large instable poles. LM method extract D, Y^{∞} matrix for maintaining the stability of the model. It follows these steps:

1) Input interpolation data to obtain $\mathbb{L}$ and $\sigma\mathbb{L}$.

$$\{\lambda = s_i, -s_i; R = I_P, I_P; W = Y^{(i)}I_P, \bar{Y}^{(i)}I_P\}$$
$$\{\mu = s_{i+1}, -s_{i+1}; L = I_P, I_P; V = Y^{(i+1)}I_P, \bar{Y}^{(i+1)}I_P\} \quad (3.114)$$

   Where $i = 1,3,\cdots,n-1$ and $\bar{Y}^{(i)}$ is the conjugate complex value of $Y^{(i)}$.

2) Obtain the rank m from SVD and calculate E, A, B, C state space matrices from (3.115).

3) From remained values from Y_s and X_s, construct eigenvectors as V_l and V_r. From these two eigenvectors, obtain the orthnormal bases Q_l and Q_r

$$Q_l = qr([Re(V_l) \quad Im(V_l) \quad 0]); Q_r = qr([Re(V_r) \quad Im(V_r) \quad 0]). \quad (3.116)$$

4) Calculate modified state space matrices:

$$\hat{E} = Q_l^T E Q_r; \hat{A} = Q_l^T A Q_r; \hat{B} = Q_l^T B; \hat{C} = C Q_r. \quad (3.117)$$

5) Extract D, Y^{∞}

$$D + sY^{\infty} = avg\{C(sE - A)^{-1}B - \hat{C}(s\hat{E} - \hat{A})^{-1}\hat{B}\}. \quad (3.118)$$

This proposed algorithm shows significant improvement on distributed network and it simulation obtains low errors on low frequencies bandwidth. With D, Y^{∞}, both the stability and passivity of time-domain model are improved.

Chapter 4 SPICE Compatible Model from State Space Matrices Model

This Chapter proposes "Sparse Matrices Algorithm" that can generate the minimum SPICE compatible models from Y-parameter using LM method. During the process of LM method, time-domain model or state space matrices are firstly obtained, which are mentioned in Section 3.4. This Chapter will introduce rest of this algorithm. Section 4.1 introduced the general method to extract SPICE compatible model from state space matrices. Section 4.2 proposes Sparse Matrices Algorithm.

4.1 General method to extract SPICE Netlist from State-space matrices

In Appendix I, we have introduced state space matrices model for some fitting method. They can be obtained from Loewner matrix method and other fitting system. This section would present the method which directly generates SPICE compatible netlist from these matrices. We remember expression of state space matrices model in (4.1).

$$\dot{x}(t) = \boldsymbol{A}x(t) + \boldsymbol{B}u(t)$$
$$y(t) = \boldsymbol{C}x(t) + \boldsymbol{D}u(t) + \boldsymbol{Y}^{\infty}\dot{u}(t) \quad (4.1)$$

In (4.1), $u(t)$ is the vector of port voltages; $y(t)$ is the vector of port current; $x(t)$ is the vector of internal statement of this system. $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}, \boldsymbol{Y}^{\infty}$ are state space matrices. Suppose P is the port number and q is the number of statement; sizes of these matrices are

$$\boldsymbol{A} \in \mathbb{R}^{q\times q}, \boldsymbol{B} \in \mathbb{R}^{q\times P}, \boldsymbol{C} \in \mathbb{R}^{P\times q}, \boldsymbol{D} \in \mathbb{R}^{P\times P}, \boldsymbol{Y}^{\infty} \in \mathbb{R}^{P\times P} \quad (4.2)$$

Some fitting methods would generate approximation model with $\boldsymbol{D} = 0$ or $\boldsymbol{Y}^{\infty} = 0$. It has little effect on generation of netlist but might influence the accuracy of the model. It would be proven in later sections.

If Loewner Matrices method is applied, then $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}, \boldsymbol{Y}^{\infty}$ are full matrices and $\boldsymbol{D}, \boldsymbol{Y}^{\infty}$ are usually positive definite matrices [47].

As it mentioned in Figure 2.1, the next process is to write these matrices into MNA equation. To demonstrate this process, the example in Section 2.5 is reused here. Figure 4.1represents the circuit; we suppose that the state space matrices $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}, \boldsymbol{Y}^{\infty}$ for this circuit are obtained through Y-parameter of the block part.

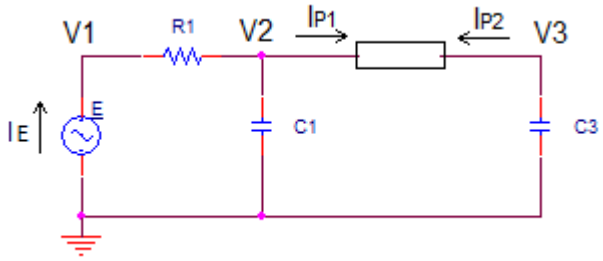


Figure 4.1 Examples for embedding state space matrices into MNA equation

Review the Section 2.5, the MNA equation for this circuit without regarding the internal structure of block part is:

$$\begin{bmatrix} g_1 & -g_1 & 0 & 1 \\ -g_1 & g_1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ I_E \end{bmatrix} + s \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & C_1 & 0 & 0 \\ 0 & 0 & C_3 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ I_E \end{bmatrix} = \begin{bmatrix} 0 \\ -I_{P1} \\ -I_{P2} \\ E \end{bmatrix} \quad (4.3)$$

Now we assume some particular data for state space matrices for better explaining the embedding process. Suppose there are r elements in $x(t)$, so in frequency domain:

$$x = [x_1 \quad x_2 \quad \cdots \quad x_r]^T \quad (4.4)$$

The block circuit in this circuit is clearly a 2-port system, thus equations in (4.1) can be rewritten as (4.5) and (4.6):

$$s \begin{bmatrix} x_1 \\ \vdots \\ x_r \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & a_{1r} \\ \vdots & \ddots & \vdots \\ a_{r1} & \cdots & a_{rr} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_r \end{bmatrix} + \begin{bmatrix} b_{11} & b_{12} \\ \vdots & \vdots \\ b_{r1} & b_{r2} \end{bmatrix} \begin{bmatrix} V_2 \\ V_3 \end{bmatrix} \quad (4.5)$$

$$\begin{bmatrix} I_{P1} \\ I_{P2} \end{bmatrix} = \begin{bmatrix} c_{11} & \cdots & c_{1r} \\ c_{21} & \cdots & c_{2r} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_r \end{bmatrix} + \begin{bmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{bmatrix} \begin{bmatrix} V_2 \\ V_3 \end{bmatrix} + s \begin{bmatrix} y_{11}^{\infty} & y_{12}^{\infty} \\ y_{21}^{\infty} & y_{22}^{\infty} \end{bmatrix} \begin{bmatrix} V_2 \\ V_3 \end{bmatrix} \quad (4.6)$$

To embed equations (4.5) and (4.6) into (4.4), we first substitute (4.6) into I_{P1} and I_{P2} in (4.3). If we move all the parts on the right side, except E, into the left side, we obtain an equation with parameter x:

$$\begin{bmatrix} g_1 & -g_1 & 0 & 1 & 0 & \cdots & 0 \\ -g_1 & g_1+d_{11} & d_{12} & 0 & c_{11} & \cdots & c_{1r} \\ 0 & d_{21} & d_{22} & 0 & c_{21} & \cdots & c_{2r} \\ 1 & 0 & 0 & 0 & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ I_E \\ x_1 \\ \vdots \\ x_r \end{bmatrix} + s \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & C_1+y_{11}^{\infty} & y_{12}^{\infty} & 0 & 0 & \cdots & 0 \\ 0 & y_{21}^{\infty} & C_3+y_{22}^{\infty} & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ I_E \\ x_1 \\ \vdots \\ x_r \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ E \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (4.7)$$

The next step is to complete the equation for parameters in x. We add (4.6) to the bottom of matrices in (4.7). Now we obtain an integrated MNA equation for whole circuit in Figure 4.1.

$$\begin{bmatrix} g_1 & -g_1 & 0 & 1 & 0 & \cdots & 0 \\ -g_1 & g_1+d_{11} & d_{12} & 0 & c_{11} & \cdots & c_{1r} \\ 0 & d_{21} & d_{22} & 0 & c_{21} & \cdots & c_{2r} \\ 1 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & -b_{11} & -b_{12} & 0 & -a_{11} & \cdots & -a_{1r} \\ \vdots & \vdots & \vdots & 0 & \vdots & \ddots & \vdots \\ 0 & -b_{r1} & -b_{r2} & 0 & -a_{r1} & \cdots & -a_{rr} \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ I_E \\ x_1 \\ \vdots \\ x_r \end{bmatrix} + s \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & C_1+y_{11}^{\infty} & y_{12}^{\infty} & 0 & 0 & \cdots & 0 \\ 0 & y_{21}^{\infty} & C_3+y_{22}^{\infty} & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 1 & \ddots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ I_E \\ x_1 \\ \vdots \\ x_r \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ E \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (4.8)$$

This MNA equation is quite simple and direct includes the contents in the block system.

To establish the prototype for this process, we conclude the MNA equation in (4.9), which only contains the nodes voltage and r statements in x:

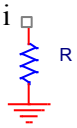
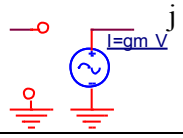
$$\begin{bmatrix} \boldsymbol{\Psi_{vv}} & \boldsymbol{\Psi_{vu}} & \mathbf{0} \\ \boldsymbol{\Psi_{uv}} & \boldsymbol{\Psi_{uu}+D} & \boldsymbol{C} \\ \mathbf{0} & \boldsymbol{-B} & \boldsymbol{-A} \end{bmatrix} \begin{bmatrix} v \\ u \\ x \end{bmatrix} + s \begin{bmatrix} \boldsymbol{\Phi_{vv}} & \boldsymbol{\Phi_{vu}} & \mathbf{0} \\ \boldsymbol{\Phi_{uv}} & \boldsymbol{\Phi_{uu}+Y^{\infty}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{I_r} \end{bmatrix} \begin{bmatrix} v \\ u \\ x \end{bmatrix} = \begin{bmatrix} \boldsymbol{J_v} \\ \boldsymbol{J_u} \\ 0 \end{bmatrix};$$

$$\boldsymbol{\overline{G}\overline{x}} + s\boldsymbol{\overline{C}\overline{x}} = \boldsymbol{\overline{b}}. \quad (4.9)$$

In equations (4.9), u is a vector of port voltages of block system; x is a vector of statements inside block or a vector of voltages of nodes inside the block; v is a vector of voltages of nodes which are belonged to the system but rather than those in u. So v plus u are the voltages of all nodes in the circuit. $\boldsymbol{\Psi}$ and $\boldsymbol{\Phi}$ present the relationships for the whole circuit without block. $\boldsymbol{J}$ present sources on circuit nodes.

To generate SPICE netlist from MNA equation in (4.9), stamps in Table 2-1 are used to quickly address the components to the data in $\boldsymbol{GC}$ matrices. In order to quickly adapt stamp, we slightly modify some stamps in table 4-1. This table only consider the generation for state space matrices; since $\boldsymbol{\Psi}$ and $\boldsymbol{\Phi}$ are generated from existing components of original circuit.

Table 4-1Modified Stamps for MNA equation

Element	Symbol	Stamp in MNA equaiton	Equation
Resistor R to ground		$\boldsymbol{G}$: $col\ i$ $row\ i$ $[1/R]$	$V_i/R = I_i$
Capacitor C to ground		$\boldsymbol{C}$: $col\ i$ $row\ i$ $[C]$	$sCV_i = I_i$
VCCS to ground		$\boldsymbol{G}$: $col\ i$ $row\ j$ $[gm]$	$gmV_i = I_j$

To generate the SPICE netlist from $\overline{\boldsymbol{G}}$ and $\overline{\boldsymbol{C}}$, we can follow the rules below:

1) Netlist Generation for $\overline{\boldsymbol{G}}$

   Components from $\overline{\boldsymbol{G}}$ are divided into two types:

   - Each data on the diagonal entry, $\overline{\boldsymbol{G}}_{ii}$, would form a resistor from node i to the ground
   - Each data not on the diagonal entry, $\overline{\boldsymbol{G}}_{ij}(i \neq j)$, would form a VCCS between node i and j to the ground.

2) Netlist Generation for $\overline{\boldsymbol{C}}$

   $\overline{\boldsymbol{C}}$ is a sparse matrix; netlist generation for $\overline{\boldsymbol{C}}$ is divided into two parts:

   - If Y^{∞} part is not null matrix, it would generate capacitors. Detailed algorithm refers to the "netlist generation for Y^{∞}" in Section 4.2 part b) .
   - The rest part of $\overline{\boldsymbol{C}}$ is an identity matrix $\boldsymbol{I}$, so they would generate capacitors with value of 1Farad between corresponding nodes to ground.

For better understanding this method, the example in is used here, pole-residue approximation can be transferred into state space matrices through methods in Appendix I.

$$y(s) = \frac{1}{s+1} + \frac{1+i}{s+1+i} + \frac{1-i}{s+1-i} + 1 + s$$

$$\boldsymbol{A} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & -1 \\ 0 & 1 & -1 \end{bmatrix}, \boldsymbol{B} = \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix}, \boldsymbol{C} = [1 \quad 1 \quad 1], \boldsymbol{D} = [1], \boldsymbol{Y}^{\infty} = [1] \quad (4.10)$$

Then matrices for MNA equations can be constructed based on (4.9) :

$$\bar{G} = \begin{bmatrix} \boldsymbol{D} & \boldsymbol{C} \\ -\boldsymbol{B} & -\boldsymbol{A} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & 1 & 0 & 0 \\ -2 & 0 & 1 & 1 \\ 0 & 0 & -1 & 1 \end{bmatrix}; \bar{C} = \begin{bmatrix} \boldsymbol{Y}^{\infty} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.11)$$

Equivalent circuit can be constructed through components in Table 4-1 and is given in Figure 4.2.

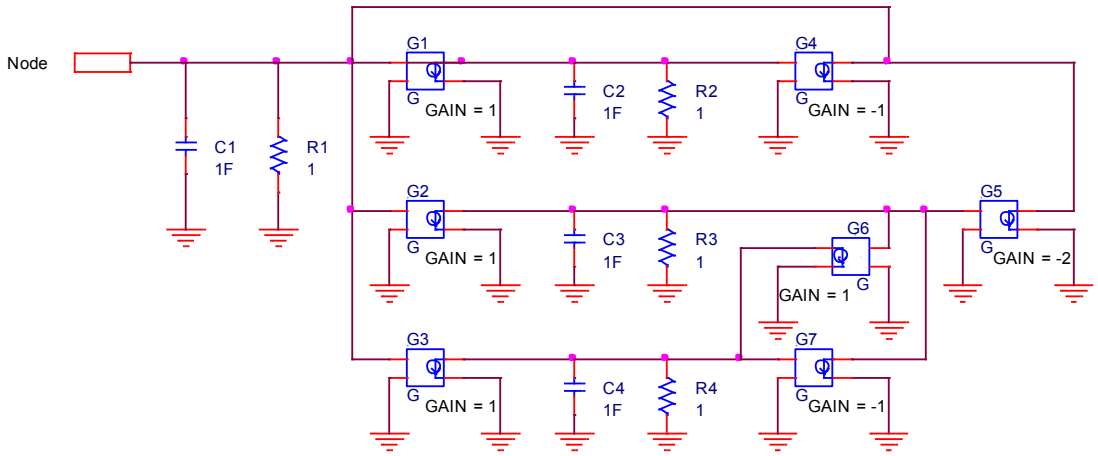


Figure 4.2 Admittance Equivalent Model from General method for state space matrices

From this result, we figure that this method is very complicated and generate so many VCCS components. The example is compared with those generated from **Sparse Matrices Algorithm**, which would be introduced in the next section.

This section described direct netlist generation method for state space matrices. However, this method would generate a large number of components for complex systems. From previous Chapters, we have noticed that state space matrices from a certain fitting method often share some properties. For example, matrix *A* through VF method is always a sparse symmetric large matrix; matrix *B* through VF method is a sparse matrix with value 0, 1 and 2 (refer to Section 3.4.1). State space matrices though LM method are always full matrices [43]. If we could apply some optimization, based on these properties, to these matrices before arranging them into MNA equation; then number of components in the result (netlist) could be significantly reduced and accuracy also could be improved. Section 4.2 will introduce the optimized algorithm on state space matrices generated from LM method.

4.2 Sparse Matrices Algorithm for State Space Matrices from LM method

State space matrices generated from LM method are shown in (4.12), the equations are in frequency domain and Y is the expression of Y-parameter. P is the port number of Y. m is the state number generated by Loewner Matrix method. We could find these equations are slightly different to (4.1). If we multiple $\boldsymbol{E}^{-1}$ to both side of first equation, then they would be same.

$$s\boldsymbol{E}x = \boldsymbol{A}x + \boldsymbol{B}u,$$
$$y = \boldsymbol{C}x + \boldsymbol{D}u + s\boldsymbol{Y}^{\infty}u;$$
$$Y = \boldsymbol{C}(s\boldsymbol{E} - \boldsymbol{A})^{-1}\boldsymbol{B} + \boldsymbol{D} + s\boldsymbol{Y}^{\infty}.$$
$$\boldsymbol{E}, \boldsymbol{A} \in \mathbb{R}^{m\times m}; \boldsymbol{B} \in \mathbb{R}^{m\times P}; \boldsymbol{C} \in \mathbb{R}^{P\times m};\ \boldsymbol{D}, \boldsymbol{Y}^{\infty} \in \mathbb{R}^{P\times P}; x \in \mathbb{C}^{m}; y, u \in \mathbb{C}^{P}. \quad (4.12)$$

These matrices have these properties:

- $\boldsymbol{E}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ have high density (or are full matrices) .
- $\boldsymbol{E}$, sometimes, is close to singular [55].
- $\boldsymbol{D}, \boldsymbol{Y}^{\infty}$ are usually positive definite [47].

If we embed these two equations into an MNA equation, it would be (4.13), similar to (4.9):

$$\begin{bmatrix} \boldsymbol{\Psi_{vv}} & \boldsymbol{\Psi_{vu}} & \boldsymbol{0} \\ \boldsymbol{\Psi_{uv}} & \boldsymbol{\Psi_{uu}} + \boldsymbol{D} & \boldsymbol{C} \\ \boldsymbol{0} & -\boldsymbol{B} & -\boldsymbol{A} \end{bmatrix} \begin{bmatrix} v \\ u \\ x \end{bmatrix} + s \begin{bmatrix} \boldsymbol{\Phi_{vv}} & \boldsymbol{\Phi_{vu}} & \boldsymbol{0} \\ \boldsymbol{\Phi_{uv}} & \boldsymbol{\Phi_{uu}} + \boldsymbol{Y}^{\infty} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{E} \end{bmatrix} \begin{bmatrix} v \\ u \\ x \end{bmatrix} = \begin{bmatrix} \boldsymbol{J_v} \\ \boldsymbol{J_u} \\ 0 \end{bmatrix} \quad (4.13)$$

If we apply direct netlist generation method in last section to (4.13); it would resulted a large number of components, since all state space matrices are full matrices. Furthermore, SPICE netlist generation for $\boldsymbol{E}$ is very difficult.

In this case, we can apply an algorithm to sparsify these state space matrices so that would generate fewer components in netlist. The idea of this algorithm is from PRIMA [8]. We acclaimed this algorithm as **Sparse Matrices Algorithm**. It can be divided into two steps: Sparse matrices generation and Netlist generation. They are introduced in the later part of this section.

a) Sparse matrices generation:

The goal of this step is to sparsify state space matrices; it would start from equations in (4.12).

1) Multiple $\boldsymbol{A^{-1}}$ on both sides of first equation, the other one remains unchanged:

$$s(\boldsymbol{A^{-1}E})x = x + (\boldsymbol{A^{-1}B})u;$$
$$y = \boldsymbol{C}x + \boldsymbol{D}u + s\boldsymbol{Y}^{\infty}u. \quad (4.14)$$

2) Sparsification of $\boldsymbol{A^{-1}E}$:

Find eigenvectors and eigenvalues for $A^{-1}E$, we assume that

$$\boldsymbol{A^{-1}E} = \boldsymbol{V\Gamma V^{-1}};\ \boldsymbol{V}, \boldsymbol{\Gamma} \in \mathbb{C}^{P\times P}. \quad (4.15)$$

Where $\boldsymbol{V}$ is eigenvectors and $\boldsymbol{\Gamma}$ is square matrix where eigenvalues are organized in diagonal order. Due to the properties of LM algorithm, elements in $\boldsymbol{\Gamma}$ are always negative for their real parts. There is an important property for netlist generation part. Suppose $x = \boldsymbol{V}\tilde{x}$, then orthogonal projection is applied on (4.15):

$$s\boldsymbol{\Gamma}\tilde{x} = \tilde{x} + \left(\boldsymbol{V^{-1}A^{-1}B}\right)u,$$
$$y = \boldsymbol{CV}\tilde{x} + \boldsymbol{D}u + s\boldsymbol{Y}^{\infty}u \quad (4.16)$$

3) Realization of equations:

After the sparsification, (4.16) are complex equations. To transform them to real equations, a transformation matrix $\boldsymbol{P}$ is used. $\boldsymbol{P} \in \mathbb{C}^{P\times P}$ is a diagonal matrix constructed as:

If $\boldsymbol{\Gamma}_n$ is real, $\boldsymbol{P}(n,n) = 1$.

If $\boldsymbol{\Gamma}_n = -a - jb$ and $\boldsymbol{\Gamma}_{n+1} = -a + jb$, $(a \geq 0, b \geq 0)$, are conjugate complex

$$\begin{bmatrix} \boldsymbol{P}_{n,n} & \boldsymbol{P}_{n,n+1} \\ \boldsymbol{P}_{n+1,n} & \boldsymbol{P}_{n+1,n+1} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ j & -j \end{bmatrix} \quad (4.17)$$

Since

$$\begin{bmatrix} \boldsymbol{P}_{n,n} & \boldsymbol{P}_{n,n+1} \\ \boldsymbol{P}_{n+1,n} & \boldsymbol{P}_{n+1,n+1} \end{bmatrix}^{-1} = \begin{bmatrix} 0.5 & -0.5j \\ 0.5 & 0.5j \end{bmatrix} \quad (4.18)$$

It would transform the complex part into real part by:

$$\begin{bmatrix} 1 & 1 \\ j & -j \end{bmatrix}\begin{bmatrix} -a - jb & 0 \\ 0 & -a + jb \end{bmatrix}\begin{bmatrix} 0.5 & -0.5j \\ 0.5 & 0.5j \end{bmatrix} = \begin{bmatrix} -a & -b \\ b & -a \end{bmatrix} \quad (4.19)$$

In this case, we assume:

$$\overline{\boldsymbol{\Gamma}} = \boldsymbol{P\Gamma P^{-1}} \quad (4.20)$$

will be a real block diagonal matrix.

In $\boldsymbol{V}$ matrix, the corresponding eigenvectors for conjugate complex $\boldsymbol{\Gamma}_n$ and $\boldsymbol{\Gamma}_{n+1}$ are $[v_c \quad v_c^*]$, since

$$[v_c \quad v_c^*]\begin{bmatrix}0.5 & -0.5j\\ 0.5 & 0.5j\end{bmatrix} = [Re(v_c) \quad Im(v_c)] \text{ (4.21)}$$

Then

$$\overline{\boldsymbol{V}} = \boldsymbol{V}\boldsymbol{P}^{-1} \text{ (4.22)}$$

will be a real matrix.

Suppose $\bar{x} = P\tilde{x}$, substitute $\bar{x}$ to (4.16) we obtain:

$$s\overline{\boldsymbol{\Gamma}}\bar{x} = \bar{x} + \overline{\boldsymbol{V}}^{-1}\boldsymbol{A}^{-1}\boldsymbol{B}u;$$
$$y = \boldsymbol{C}\overline{\boldsymbol{V}}\bar{x} + \boldsymbol{D}u + s\boldsymbol{Y}^{\infty}u. \text{ (4.23)}$$

Now all matrices in (4.23) are sparse real matrices; rewrite them into format of (4.12):

$$s\boldsymbol{E}_{\boldsymbol{s}}\bar{x} = \boldsymbol{A}_{\boldsymbol{s}}\bar{x} + \boldsymbol{B}_{\boldsymbol{s}}u;$$
$$y = \boldsymbol{C}_{\boldsymbol{s}}\bar{x} + \boldsymbol{D}u + s\boldsymbol{Y}^{\infty}u. \text{ (4.24)}$$

Where $\boldsymbol{E}_{\boldsymbol{s}} = \overline{\boldsymbol{\Gamma}}, \boldsymbol{A}_{\boldsymbol{s}} = \boldsymbol{I}_{\boldsymbol{m}}, \boldsymbol{B}_{\boldsymbol{s}} = \overline{\boldsymbol{V}}^{-1}\boldsymbol{A}^{-1}\boldsymbol{B}, \boldsymbol{C}_{\boldsymbol{s}} = \boldsymbol{C}\overline{\boldsymbol{V}}$.

$\boldsymbol{E}_{\boldsymbol{s}}$ is real sparse matrix with diagonal block values. $\boldsymbol{A}_{\boldsymbol{s}}$ is an identity matrix. Compare them to $\boldsymbol{E}$ and $\boldsymbol{A}$ in (4.13), they have achieved excellent simplification.The matrices $\boldsymbol{B}_{\boldsymbol{s}}$ and $\boldsymbol{C}_{\boldsymbol{s}}$ are still dense; however, they are the multipliers for u and $\bar{x}$; so their sparsities are less concerned. The matrices $\boldsymbol{D}$ and $\boldsymbol{Y}^{\infty}$ are unchanged.

b) Netlist generation

Before fitting state space matrices in (4.24) into MNA equation, a slight transformation need to be applied on $\boldsymbol{E}_{\boldsymbol{s}}$ so transformed MNA equation can better adapt to capacitor stamp in later part of this section.

Construct diagonal matrix $\boldsymbol{Q} \in \mathbb{R}^{m\times m}$ as a transformation matrix:

If $\boldsymbol{E}_s(n,n)$ is not part of diagonal block, $\boldsymbol{Q}(n,n) = 1$.

If $\begin{bmatrix}\boldsymbol{E}_{s_{n,n}} & \boldsymbol{E}_{s_{n,n+1}}\\ \boldsymbol{E}_{s_{n+1,n}} & \boldsymbol{E}_{s_{n+1,n+1}}\end{bmatrix} = \begin{bmatrix}-a & -b\\ b & -a\end{bmatrix}, \begin{bmatrix}\boldsymbol{Q}_{n,n} & \boldsymbol{Q}_{n,n+1}\\ \boldsymbol{Q}_{n+1,n} & \boldsymbol{Q}_{n+1,n+1}\end{bmatrix} = \begin{bmatrix}-1 & 0\\ 0 & 1\end{bmatrix}$. (4.19)

Multiple $\boldsymbol{Q}$ to the first equation of (4.18), we obtain new equations:

$$s\boldsymbol{Q}\boldsymbol{E}_{\boldsymbol{s}}\bar{x} = \boldsymbol{Q}\bar{x} + \boldsymbol{Q}\boldsymbol{B}_{\boldsymbol{s}}u,$$
$$y = \boldsymbol{C}_{\boldsymbol{s}}\bar{x} + \boldsymbol{D}u + s\boldsymbol{Y}^{\infty}u \text{ (4.20)}$$

Write (4.20) into MNA equation we can obtain (4.21), since the diagonal element in $\boldsymbol{E_s}$ is always negative(see (4.19)). This equation is slightly different to (4.13).

$$\begin{bmatrix} \boldsymbol{\Psi_{vv}} & \boldsymbol{\Psi_{vu}} & \boldsymbol{0} \\ \boldsymbol{\Psi_{uv}} & \boldsymbol{\Psi_{uu}+D} & \boldsymbol{C_s} \\ \boldsymbol{0} & \boldsymbol{QB_s} & \boldsymbol{Q} \end{bmatrix} \begin{bmatrix} v \\ u \\ \bar{x} \end{bmatrix} + s \begin{bmatrix} \boldsymbol{\Phi_{vv}} & \boldsymbol{\Phi_{vu}} & \boldsymbol{0} \\ \boldsymbol{\Phi_{uv}} & \boldsymbol{\Phi_{uu}+Y^{\infty}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{-QE_s} \end{bmatrix} \begin{bmatrix} v \\ u \\ \bar{x} \end{bmatrix} = \begin{bmatrix} \boldsymbol{J_v} \\ \boldsymbol{J_u} \\ 0 \end{bmatrix};$$

$$\overline{\boldsymbol{G}}\overline{\boldsymbol{x}} + s\overline{\boldsymbol{C}}\overline{\boldsymbol{x}} = \overline{\boldsymbol{b}}. \quad (4.21)$$

Now we can start to extract components to form SPICE netlist. Similar to Section 4.1, only state space matrices are discussed here. The processes are again divided into two parts: Netlist Generation for $\overline{\boldsymbol{G}}$ and Netlist Generation for $\overline{\boldsymbol{C}}$. The stamps for this method are resistors, capacitors and VCCS stamps, which are reviewed in Table 2-1 and 4-1.

1) Netlist Generation for $\boldsymbol{G}$

   There are four parts in $\boldsymbol{G}$: $\boldsymbol{D}, \boldsymbol{C_s}, \boldsymbol{QB_s}, \boldsymbol{Q}$.

   i. $\boldsymbol{Q}$ is diagonal matrix contains only $+1$ and -1. The netlist for this part will be resistors from corresponding node to ground.

   ii. Matrices $\boldsymbol{QB_s}$ and $\boldsymbol{C_s}$ are full matrices and data in each entry are different. VCCS stamp to ground (Table 4-1) is used so that every value in them represents a VCCS in SPICE netlist.

   iii. Matrix $\boldsymbol{D}$ is a positive definite; it can be divided into $P(P-1)/2$ resistor stamps between ports and P resistor stamps between port and ground.

   An example is given for $P = 3$:

$$D(:,:) = \begin{bmatrix} a & d & e \\ d & b & f \\ e & f & c \end{bmatrix} = \begin{bmatrix} -d & d & 0 \\ d & -d & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} -e & 0 & e \\ 0 & 0 & 0 \\ e & 0 & -e \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & -f & f \\ 0 & f & -f \end{bmatrix} + \begin{bmatrix} a+d+e & 0 & 0 \\ 0 & b+d+f & 0 \\ 0 & 0 & c+e+f \end{bmatrix} \quad (4.22)$$

   The capacitors are:

   $C_1 = -d$ between port 1 and 2; $C_2 = -e$ between port 1 and 3; $C_3 = -f$ between port 2 and 3.

   $C_4 = a + d + e$ between port 1 and ground; $C_4 = b + d + f$ between port 2 and ground; $C_4 = c + e + f$ between port 3 and ground.

2) Netlist Generation for $\boldsymbol{C}$

There are two parts in $\boldsymbol{C}$: $\boldsymbol{Y^{\infty}}, -\boldsymbol{QE_s}$.

i. Similar to matrix $\boldsymbol{D}$, matrix $\boldsymbol{Y^{\infty}}$ can be decomposed into $P(P-1)/2$ capacitor stamps between the ports and P capacitor stamps between port and ground.

ii. Matrix $-\boldsymbol{QE_s}$ is combination of diagonal entries, which corresponding to real eigenvalue, and block diagonal entries which corresponding to complex conjugate eigenvalues.

- For non-zero diagonal entries, capacitor stamp between port to ground is used.
- For block diagonal entries, this block can be divided into three capacitor stamps:

$$\begin{bmatrix} -\boldsymbol{QE}_{s_{n,n}} & -\boldsymbol{QE}_{s_{n,n+1}} \\ -\boldsymbol{QE}_{s_{n+1,n}} & -\boldsymbol{QE}_{s_{n+1,n+1}} \end{bmatrix} = \begin{bmatrix} -a & -b \\ -b & a \end{bmatrix}$$

$$= \begin{bmatrix} b & -b \\ -b & b \end{bmatrix} + \begin{bmatrix} -a-b & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & a-b \end{bmatrix} \quad (4.23)$$

So $C_1 = b$ between node n and $n+1$; $C_2 = -a-b$ between node n and ground; $C_3 = a-b$ between node $n+1$ and ground.

If Matrix $-\boldsymbol{E_s}$ was used instead of $-\boldsymbol{QE_s}$ here, the netlist would be much more complicated.

Sparse Matrices Algorithm is presented above to generate SPICE compatible model (netlist) from state space matrices. Compare to the method in previous section, this method uses the properties of state space matrices and create a new approach to achieve fewer components and better efficiency. The larger number of ports system has, the more superiority on macromodelling method would present. The result would be shown in the rest in Chapter 6.

To demonstrate this algorithm, the example in last section is reused. At first, we input $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ into sparse matrices generation (part a) to obtain sparsified matrices (we use $\boldsymbol{E}$ as a rank 3 identity matrix). $\boldsymbol{D}$ and $\boldsymbol{Y^{\infty}}$ don't change. sparsified matrices are shown below:

$$\boldsymbol{E_s} = \begin{bmatrix} -0.5 & -0.5 & 0 \\ 0.5 & -0.5 & 0 \\ 0 & 0 & -1 \end{bmatrix}, \boldsymbol{B_s} = \begin{bmatrix} \sqrt{2} \\ \sqrt{2} \\ -1 \end{bmatrix}, \boldsymbol{C_s} = \begin{bmatrix} -\frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & 1 \end{bmatrix} \quad (4.24)$$

Based on $\boldsymbol{E_s}$, transformation matrix $\boldsymbol{Q}$ is

$$\boldsymbol{Q} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (4.25)$$

Thus matrices for MNA equations are in (4.26):

$$\overline{\boldsymbol{G}} = \begin{bmatrix} \boldsymbol{D} & \boldsymbol{C_s} \\ \boldsymbol{QB_s} & \boldsymbol{QA_s} \end{bmatrix} = \begin{bmatrix} 1 & -\frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & 1 \\ -\sqrt{2} & -1 & 0 & 0 \\ \sqrt{2} & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{bmatrix},$$

$$\overline{\boldsymbol{C}} = \begin{bmatrix} \boldsymbol{Y}^{\infty} & 0 \\ 0 & -\boldsymbol{QE_s} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -0.5 & -0.5 & 0 \\ 0 & -0.5 & 0.5 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \quad (4.26)$$

The matrix $\overline{\boldsymbol{C}}$ can be transferred into (4.27) so it generates 4 capacitors:

$$\overline{\boldsymbol{C}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0.5 & -0.5 & 0 \\ 0 & -0.5 & 0.5 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.27)$$

The equivalent model is given in Figure 4.3.

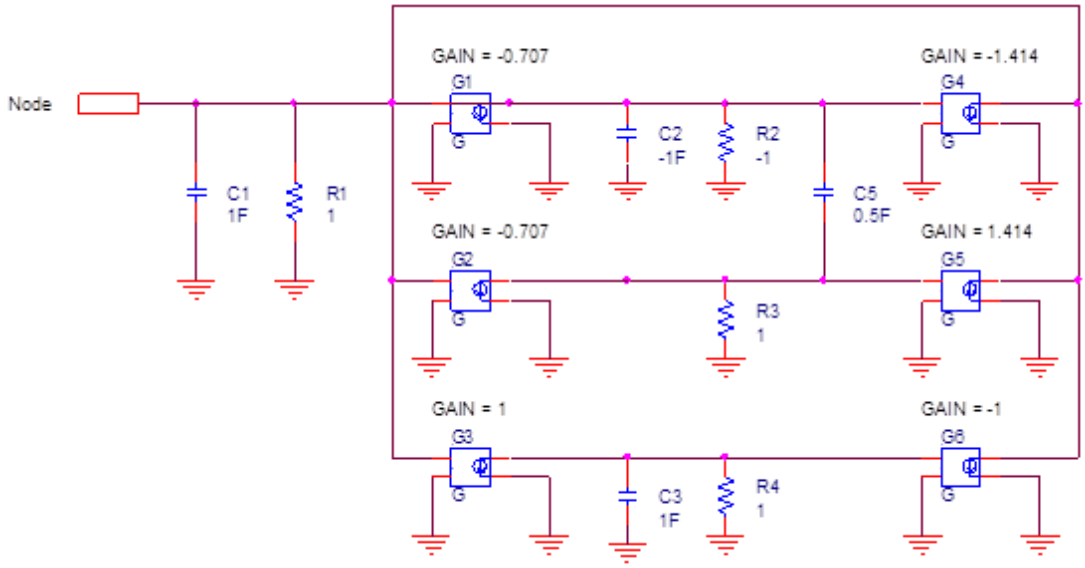


Figure 4.3 Admittance Equivalent Model from Sparse matrices algorithm

To comment two methods in this chapter, method from Section 4.1 generates large number of components, compared to Sparse Matrices Algorithm (SMA). In the simulation part in Chapter 6, we will further analyze the efficiency and accuracy of **SMA**.

The drawback of SMA is that there are some negative components, such as -1Ω resistors or -1F capacitors, in the netlists. Since these components don't exist in real application, they might have uncertain effects on simulation stability. Several methods are reviewed and proposed in next chapter to minimize the number of negative components.

Chapter 5 SPICE Compatible Model from Pole-Residue Model

In this chapter, Foster's network synthesis [59] and Otto Brune's method [56-58] on transfer impedance are reviewed. Their ideas provide a new approach for deriving networks based on transfer functions, which could be pole-residue or rational approximations. Section 5.1 introduces Positive-real (PR) property of function of frequency. Its property is a very important concept on SPICE netlist macromodeling for rational functions. Section 5.2 describes Equivalent admittance model on Y-parameter, this step divides the multiport system into numbers of one-port admittance problems. Section 5.3 reviews Foster's network synthesis and proposes Modified Foster's Method based on Foster-like circuit. Section 5.4 reviews Brune's method. Section 5.5 proposes Foster-Brune Method, which combines Modified Foster's Method and Brune's method. The detailed analysis and proof for these methods are recored in Appendix II~IV.

5.1 Positive Real property on frequency transfer functions

Before starting the algorithms, Positive-real (PR) rational functions, an important property for deriving transfer networks, must be introduced at first. Suppose a function of frequency:

$$F(s) = R(\sigma, \omega) + jX(\sigma, \omega), s = \sigma + j\omega. \text{ (5.1)}$$

$F(s)$ is a Positive-real (PR) function if [69]:

A. $F(s)$ is analytic in the right half plane.
B. Its real part $R(\sigma, \omega)$ is nonnegative on the j-axis.
C. Any j-axis poles are simple and have nonnegative residues

One testing method for PR function is given in [56]:

$$F(s) = \frac{a_m s^m + a_{m-1} s^{m-1} + a_{m-2} s^{m-2} + \cdots + a_0}{b_n s^n + b_{n-1} s^{n-1} + b_{n-2} s^{n-2} + \cdots + b_0} = \frac{P(s)}{Q(s)} = \frac{M_1 + N_1}{M_2 + N_2} \text{ (5.2)}$$

Where M denotes the real part of $P(s)$ or $Q(s)$, N denotes the imaginary part of $P(s)$ or $Q(s)$, suppose m is an even number and also for below:

$$M_1(s) = a_0 + a_2 s^2 + \cdots + a_{m-2} s^{m-2} + a_m s^m;$$

$$N_1(s) = a_1 s + a_3 s^3 + \cdots + a_{m-3} s^{m-3} + a_{m-1} s^{m-1}. \text{ (5.3)}$$

Then real part $R(\sigma, \omega)$ of $F(s)$ is:

$$Re[F(j\omega)] = \left(\frac{M_1 M_2 - N_1 N_2}{M_2^2 - N_2^2}\right)_{s=j\omega} \text{ (5.4)}$$

Condition B can transfer to (5.5):

$$A(\omega^2) = (M_1 M_2 - N_1 N_2)_{s=j\omega} \geq 0 \text{ (5.5)}$$

Condition A and C can transfer to $P(s) + Q(s)$ is Hurwitz [56].

In [56], simple testing conditions for (5.2) are introduced as:

- All $a_m, a_{m-1}, \cdots, a_0$ and $b_n, b_{n-1}, \cdots, b_0$ are positive real numbers, only a_m, a_0, b_n, b_0 can be zero.
- All poles and zeros of $F(s)$ lie on left half plane or on imaginary axis of s-plane

In [70], some others theorems are introduced for testing PR property of the function. Appendix IV has discussed PR conditions of some specific rational functions used in later sections.

5.2 Equivalent admittance model from Y parameter

This section introduced method to separate Y-parameter into number of one-port systems. After having applying VF method on these one-port systems, we could apply Brune's method and other methods, proposed in this chapter, on pole-residue set. Figure 5.1 introduces flowchart of the whole process, from Y-parameter to SPICE compatible macromodel, and we can figure out that this process is the first step of whole method.

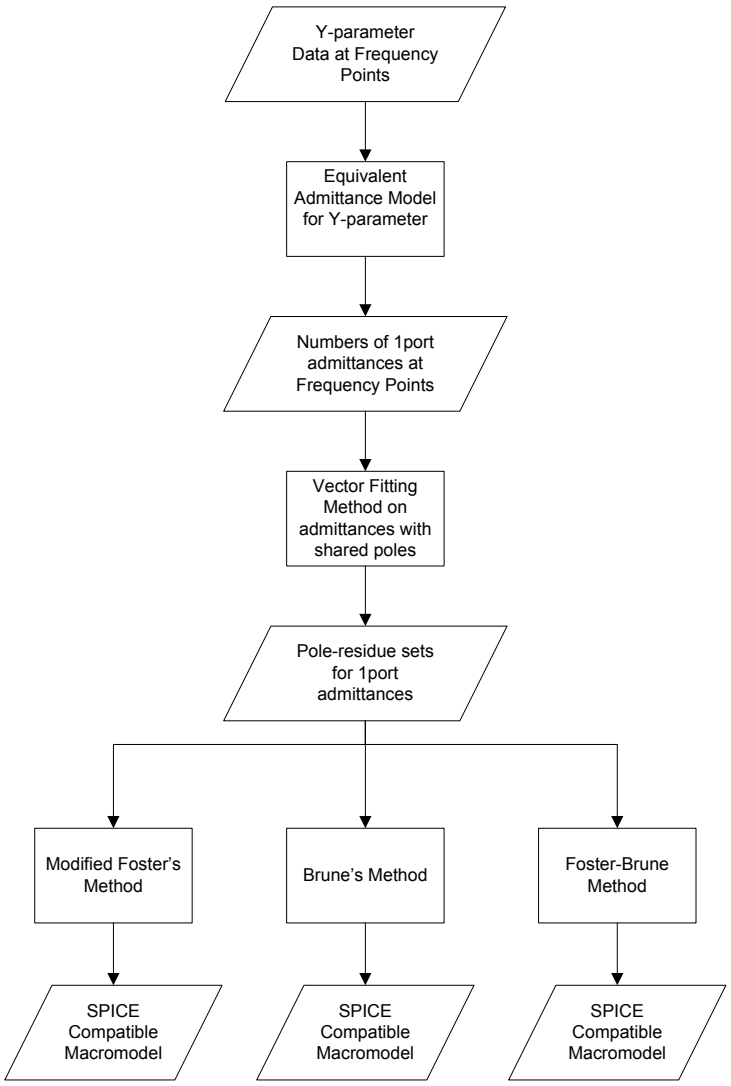


Figure 5.1 Flowchart for SPICE netlist Generation by VF method

Here we start equivalent admittance model for Y-parameter. Based on the definition of Y-parameter, port-to-port admittances of an n-port system can be calculated from its Y-matrix:

$$Y = \begin{bmatrix} Y_{11} & Y_{12} & \cdots & Y_{1n} \\ Y_{21} & Y_{22} & \cdots & Y_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ Y_{n1} & Y_{n2} & \cdots & Y_{nn} \end{bmatrix} \Rightarrow y_{ij} = \begin{cases} Y_{ii} + \sum_{i \neq j} Y_{ij} & if\ i \neq j \\ -Y_{ij} & if\ i = j \end{cases}; i, j \in [1, n] \quad (5.6)$$

For an example, the equivalent model of a 4-port system ($n = 4$) is given in Figure 5.1:

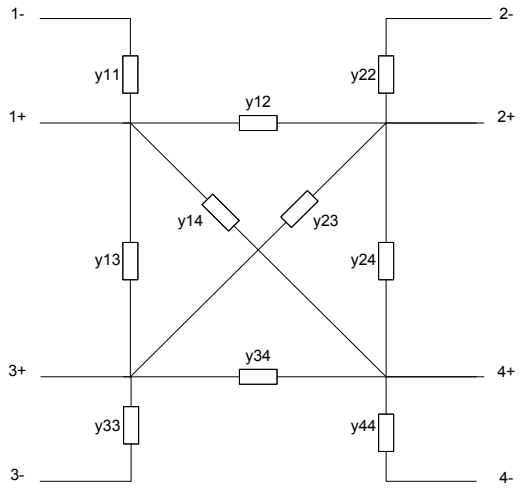


Figure 5.2 Equivalent admittance model of a 4-port system

From pole-residue model generated from VF method or LM method, the transfer function for admittance can be derived as:

$$y_{ij}(s) = \frac{r_1^{(i,j)}}{s - p_1} + \frac{r_2^{(i,j)}}{s - p_2} + \cdots + \frac{r_{N-1}^{(i,j)}}{s - p_{N-1}} + \frac{r_N^{(i,j)}}{s - p_N} + d^{(i,j)} + h^{(i,j)} s \quad (5.7)$$

In this case, SPICE compatible model for a large system are divided into number of equivalent circuits of 1-port system, which significantly reduce the complexity of modelling. For deriving equivalent circuit from 1-port system (5.7), Foster's method and Brune's method can be applied.

Later on, Section 5.3 will introduce Modified Foster's method, which generates Foster-like circuit based on Foster's network synthesis technique. Section 5.4 will review Brune's method. Section 5.5 will propose a new algorithm that combines Foster's and Brune's method.

5.3 Foster's network synthesis and Modified Foster's Method

In this section, Foster's network synthesis[58] for admittance function is used to create Foster-like circuit. This process also works for impedance function through Z-parameter of the system, but with different circuit prototypes.

Suppose (5.8) is the sum of pole-residue format rational function. Our algorithm is to write a small part $y_f(s)$ of $y(s)$ at each round. $y_f(s)$ usually consists of 1 or 2 individual rational functions and can generate a Foster-like circuit through "a reactance theorem"[58]. The rest of $y(s)$ would have less components. We acclaim this method as "Modified Foster's method" and this method can be implemented as:

$$y(s) = \frac{r_i}{s-p_i} + y'(s) = y_f(s) + y'(s) \quad (5.8)$$

Since VF method gives all poles on left-half of s-planes, $y_f(s)$ can be divided into real poles case and complex-conjugate poles case. The stamps for different cases and models are shown in Table 5-1 and Table 5-2. The detailed values for each component and proof for positivity see Appendix II.

a. If p_i is real, suppose :

$$y_f(s) = \frac{r_i}{s-p_i} = \frac{a}{s+b} \quad (5.9)$$

We can obtain different pattern of models from checking PR property of $y_f(s)$. All possible cases are given in Table 5-1, each case derive a model stamp which contain several components. There are one PR case, one non PR case and one limit case. If $y_f(s)$ is PR, then the process will be P.R. case. If $y_f(s)$ is non PR, then the process will be non PR case. Limit case only happens when some parameters in $y_f(s)$ is zero (or smaller than tolerance value). Limit cases would generate some components with negative values; however, they rarely happen in real-world simulation. Their function is to handle the critical values in simulation or to prevent the component with 0 or ∞ value.

For non PR and of limit cases, this algorithm sometimes generates a resistor G_0 parallelized the model. This resistor G_0 can switch a non PR. $y_f(s)$ to a PR $y_f'(s)$ by:

$$y_f'(s) = y_f(s) + G_0; Re\left[y_f(s) + G_0\right] \geq 0 \Leftrightarrow y_f'(s) \text{ is PR} \quad (5.10)$$

If resistor G_0 happens in the process, then it will be absorbed by parameter d of $y(s)$ in (5.7) that

$$d' = d - G_0 \quad (5.11)$$

b. p_i is a complex pole , then p_i, p_{i+1} are conjugate poles pairs. Suppose:

$$y_f(s) = \frac{r_i}{s-p_i} + \frac{r_i^*}{s-p_i^*} = \frac{a_1 s + a_0}{s^2 + b_1 s + b_0} \quad (5.12)$$

There are one PR case, two non PR cases and one four limit cases for this expression. They are given in Table 5-2.

Table 5-1 Cases and Model Stamps for Modified Foster's Method (Real Pole)

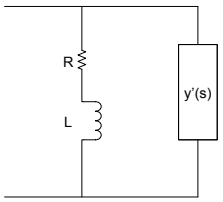
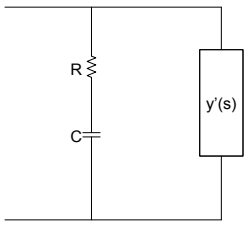
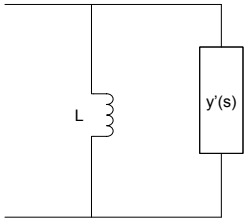
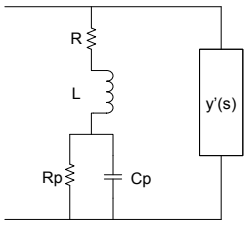
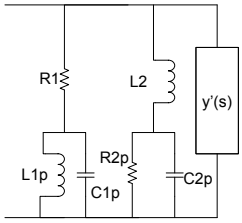
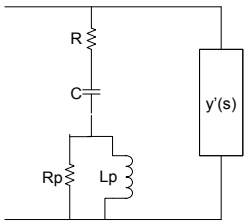
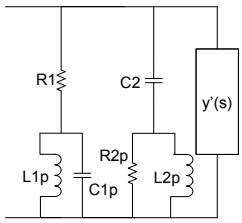
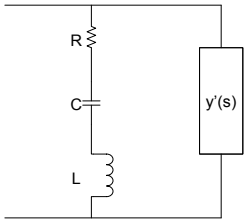
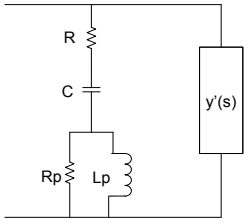
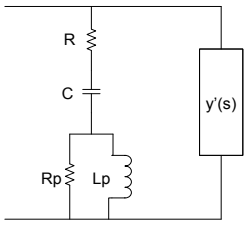
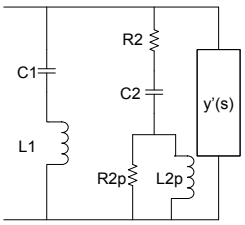
Case	Condition	y_f Model
Case 1: PR case	$a > 0, b > 0.$ $y_f(s)$ is P.R.	
Case 2:non PR case	$a < 0, b > 0$. $y_f(s)$ is non P.R. $G_0 = -\frac{a}{b}$.	
Case 3: Limit Case	$b = 0.$	

Table 5-2 Cases and Model Stamps for Modified Foster's Method (Complex Poles)

Case	Condition	y_f Model
PR case	$a_1 b_1 \geq a_0 > 0$. $y_f(s)$ is P.R.	
non PR case 1	$a_0 > 0, a_1 b_1 < a_0$. $y_f(s)$ is non P.R. $G_0 = \frac{a_0 - a_1 b_1}{b_1^2}$.	
non PR case 2	$a_0 < 0$. $y_f(s)$ is non P.R. Situation 1: $(a_1 b_0 - a_0 b_1) \geq \frac{(-a_0) b_0}{b_1} \geq 0$; $G_0 = \frac{-a_0}{b_0}$. Otherwise, Situation 2: $G_0 = \frac{(-a_0) - a_1 b_1}{b_1^2}$.	 Sit. 1  Sit. 2
Limit Case 1	$a_1 \neq 0, a_0 = 0$. $G_0 = -\frac{a_0}{b_0}$.	
Limit Case 2	$a_1 = 0, a_0 \neq 0,$ $b_1 \neq 0$. $G_0 = -\frac{a_0}{b_0}$.	

Limit Case 3	$a_1 \neq 0, a_0 \neq 0,$ $b_1 = 0.$ $G_0 = -\frac{a_0}{b_0}.$	
Limit Case 4	$a_1 = 0, a_0 \neq 0,$ $b_1 = 0$. $G_0 = -\frac{a_0}{b_0}.$	

All cases for Modified Foster Method are given above; MFM algorithm generates all positive value components for PR cases and non PR cases. For limit cases, some negative values component might be generated. Since limit cases rarely occurs, their effects can be neglect and MFM algorithm can guarantee that most of components in the model have positive value. The inference and proof are in Appendix II. The advantages for MFM algorithm are fast speed of simulation and simple circuit model.

The drawback is that d' sometimes results a negative for PR $y(s)$. The reason is that G_0 is a parameter to insure a positive $y_f'(s)$ in (5.10), it will not break PR property of $y(s)$ if

$$G_0 = -min_{s=j\omega}\, Re[y_f(s)] \quad (5.13)$$

In our algorithm, the chosen G_0 is greater or equal to this value. It means when positive d extract too much large value of G_0, d' cannot be guaranteed to be positive.

To present an example for this method, example in Chapter 4 is used again:

$$y(s) = \frac{1}{s+1} + \frac{1+i}{s+1+i} + \frac{1-i}{s+1-i} + 1 + s \quad (5.14)$$

The first step is to form ration function parts:

$$y(s) = y_{f1}(s) + y_{f2}(s) + 1 + s; y_{f1}(s) = \frac{1}{s+1}, y_{f2}(s) = \frac{2s+4}{s^2+2s+2} \quad (5.15)$$

- $y_{f1}(s)$ is real pole PR case, equivalent circuit refers to case 1 in Table 5-1.

$$y_{f1}(s) = \frac{1}{s+1} = \frac{1}{L_1 s + R_1} \quad (5.16)$$

- $y_{f2}(s)$ is complex poles PR case, equivalent circuit refers to case 1 in Table 5-2.

$$z_{f2}(s) = \frac{1}{y_{f2}(s)} = \frac{s^2+2s+2}{2s+4} = \frac{1}{2}s + \frac{1}{s+2} = L_2 s + \frac{1}{C_1 s + 1/R_2} \quad (5.17)$$

The rest parts are ending resistor and capacitor. The equivalent circuit is in Figure 5.3.

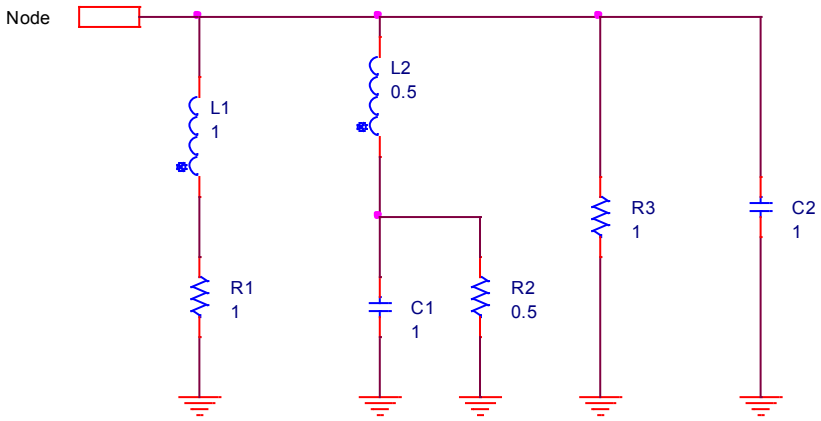


Figure 5.3 Admittance Equivalent Model from Modified Foster's method

Compare to previous methods in Section 4.1 and 4.2, Modified Foster's method could handle non PR case and generate less components for the equivalent model.

5.4 Brune's Process

Brune's method has been proposed by Otto Brune in 1931[57]. This classical method works on PR character of rational function, generates minimum and positive-value components for a driving-point impedance or admittance. Brune method treat a whole rational function as (5.18), generate circuit components from a separated small part of this function in one process and leave the rest with less order for the next. In [62-65], Brune's process is concluded into 8 cases; these cases for driving point admittance are reviewed here. Start from a PR rational function for admittance with a same order of n:

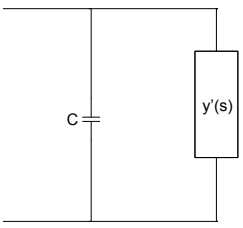
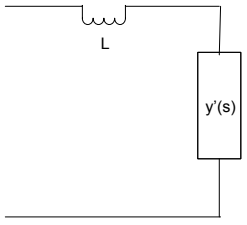
$$y(s) = \frac{a_n s^n + a_{n-1} s^{n-1} + a_{n-2} s^{n-2} + \cdots + a_0}{b_n s^n + b_{n-1} s^{n-1} + b_{n-2} s^{n-2} + \cdots + b_0} = \frac{P(s)}{Q(s)} = \frac{M_1 + N_1}{M_2 + N_2} \quad (5.18)$$

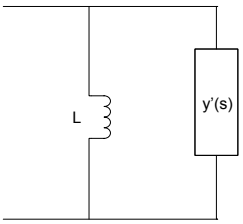
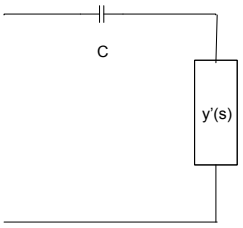
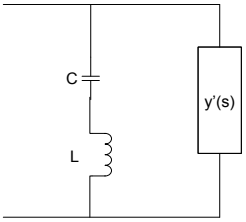
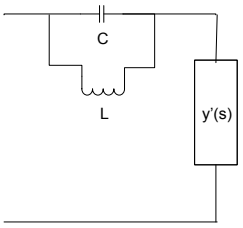
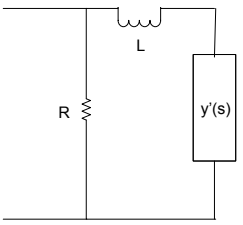
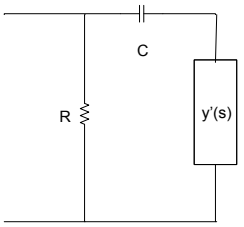
Similar to Modified Foster's method, this algorithm writes models for $y_f(s)$ at each process and leave $y'(s)$ a rational function with less order:

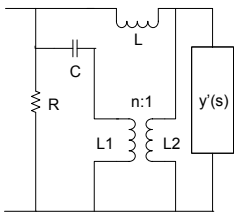
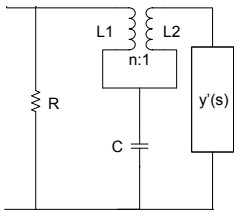
$$y(s) = y_f(s) + \frac{a'_m s^m + \cdots + a'_0}{b'_m s^m + \cdots + b'_0} = y_f(s) + y'(s) \quad (5.19)$$

The models for each case are listed in Table 5-3. More details, such as component values, are given in Appendix III.

Table 5-3 Cases and Model Stamps for Brune's Process

Case	Condition	y_f Model
Case 0	$y(s) = G$	
Case 1	$b_n = 0$	
Case 2	$a_n = 0$	

Case 3	$b_0 = 0$	
Case 4	$a_0 = 0$	
Case 5	$Q(s)\|_{s=\pm j\omega_p} = 0$, $y(s)$ has poles on imaginary axis	
Case 6	$P(s)\|_{s=\pm j\omega_p} = 0$, $y(s)$ has zeros on imaginary axis	
Case 7	$R = Re[y(s)]_{min}\|_{s=j\omega_p}$, $\omega_p \epsilon [0, \infty)$ Situation 1: $\omega_p = \infty$	
	Situation 2: $\omega_p = 0$	

	Situation 3: $y(j\omega_p)$ $= R + Im[y(j\omega_p)]$	Type 1: $Im[y(j\omega_p)] < 0$	
		Type 3: $Im[y(j\omega_p)] > 0$	

All the cases for Brune's method are introduced above. Case 0 is the end case; Case 1 to 6 is usually used only in the beginning or near the end of whole process in real world simulation. If the order n is large, most of time Brune's method are running case 7. The key advantage for Brune's method is that it generates minimum reactive elements. The drawback is that is method is only applicable for PR case. In real world simulation for n-port system, the transfer functions, sometimes, are not PR. Then Brune's method is very limited for handling them.

Another drawback of Brune's method is the overflow of limits. Simulation under high frequency must use curved data to avoid the overflow so that values of a, b will not be over simulation limits. However, the overflow still would happen when order is high (≥ 50). In this case, simulation cannot be completed.

We present the example by Brune's method here, to compare with the result from MFM in previous section. Since the example is positive real, we can apply Brune's method to it. The whole rational equation is firstly calculated. Based on cases in Table 5-3, the process is:

Step 1, case 1:

$$y(s) = \frac{s^4+4s^3+10s^2+14s+8}{s^3+3s^2+4s+2} = s + \frac{s^3+6s^2+12s+8}{s^3+3s^2+4s+2} = y_{f1}(s) + y^{(1)}(s) \quad (5.20)$$

Step 2, case 7 situation 1:

$$y_{f1}(s) = \frac{s^3+6s^2+12s+8}{s^3+3s^2+4s+2} = 1 + \frac{3s^2+8s+6}{s^3+3s^2+4s+2} = 1 + \frac{1}{\frac{\frac{1}{3}s+\frac{\frac{1}{3}s^2+2s+2}{3s^2+8s+6}}} = 1 + \frac{1}{\frac{1}{3}s+\frac{1}{y^{(2)}(s)}} \quad (5.21)$$

Step 3, case 7 situation 2:

$$y^{(2)}(s) = \frac{3s^2+8s+6}{\frac{1}{3}s^2+2s+2} = 3 + \frac{2s^2+2s}{\frac{1}{3}s^2+2s+2} = 3 + \frac{1}{\frac{1}{\frac{1}{3}s}+\frac{1}{2s+2}} = 3 + \frac{1}{\frac{1}{s}+\frac{1}{y^{(3)}(s)}} \quad (5.22)$$

Step 4, case 3:

$$y^{(3)}(s) = \frac{2s+2}{\frac{1}{3}s} = \frac{6}{s} + 6 = \frac{6}{s} + y^{(4)}(s) \quad (5.23)$$

Step 5, case 0 and process is end here:

$$y^{(4)}(s) = 6. \quad (5.24)$$

So the equivalent circuit, based on stamps in Table 5-3, is given in Figure 5.4.

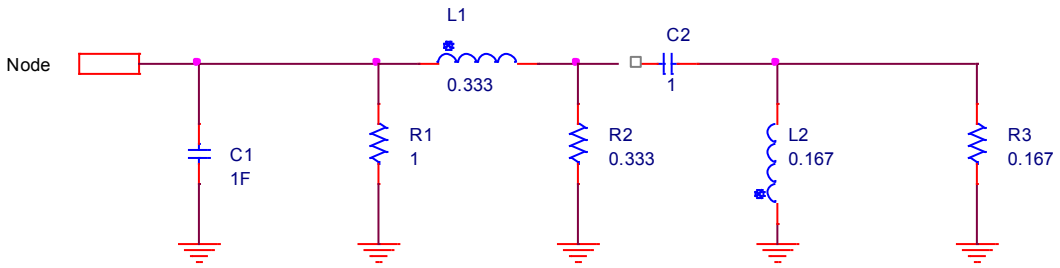


Figure 5.4 Admittance Equivalent Model from Brune's method

In the next section, a new method based on Modified Foster's method and Brune's method is proposed. This method combines Foster's method and Brune's method, shares their advantages and avoids their drawbacks.

5.5 Foster-Brune Method

In previous sections, we notice that Modified Foster's Method deals with admittance $y(s)$ as a number of small parts and Brune's process treats $y(s)$ as a whole part (rational function).In this section, a method based on Modified Foster's method and Brune's method is proposed and it can be called "Foster-Brune method". This method absorbs the advantages and avoids the disadvantages of both methods. Design purpose is shown in Figure 5.3.

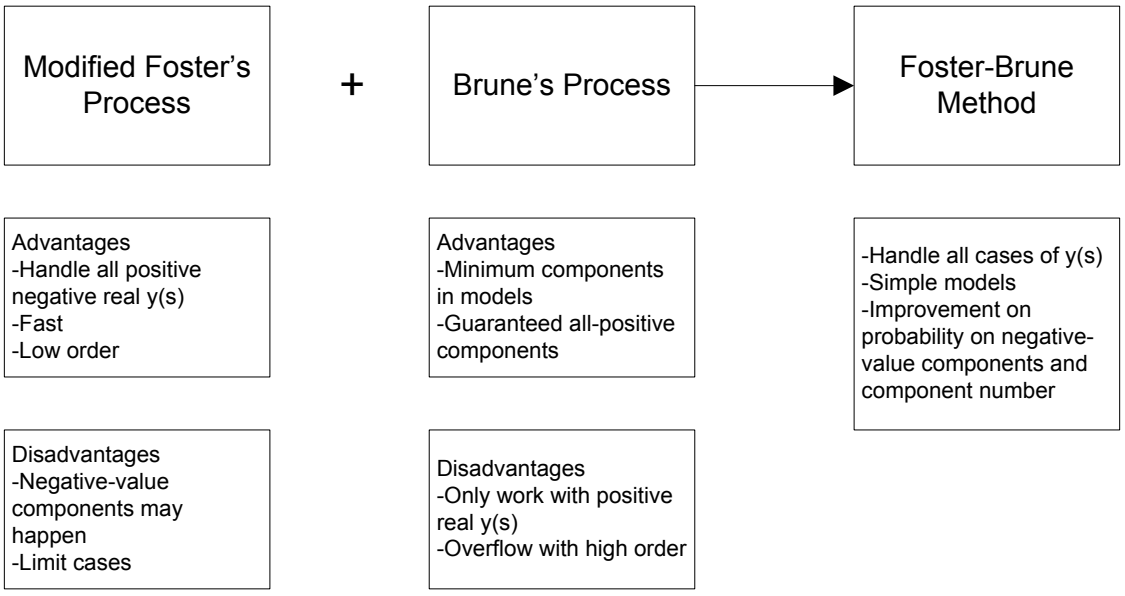


Figure 5.5 Desired properties of Foster-Brune Method

In Brune-Foster method, several pole-residue components are chosen to form a low-order PR rational function and apply Brune process on them, the rest of components are proceed by Foster process. In another word, we use Modified Foster process for general cases, but we seek the PR component for Non PR component to form a PR low-order rational function and use Brune process for these low-order functions. It follows steps below and the follow chart for FBM algorithm is shown in Figure 5.4.

1. Start from pole-residue format for admittance, separate PR and non PR parts.
2. Seek the partner for non PR parts to form low-order PR rational functions
3. Apply Brune's process on these rational functions
4. Apply Foster's process on rest pole-residue components

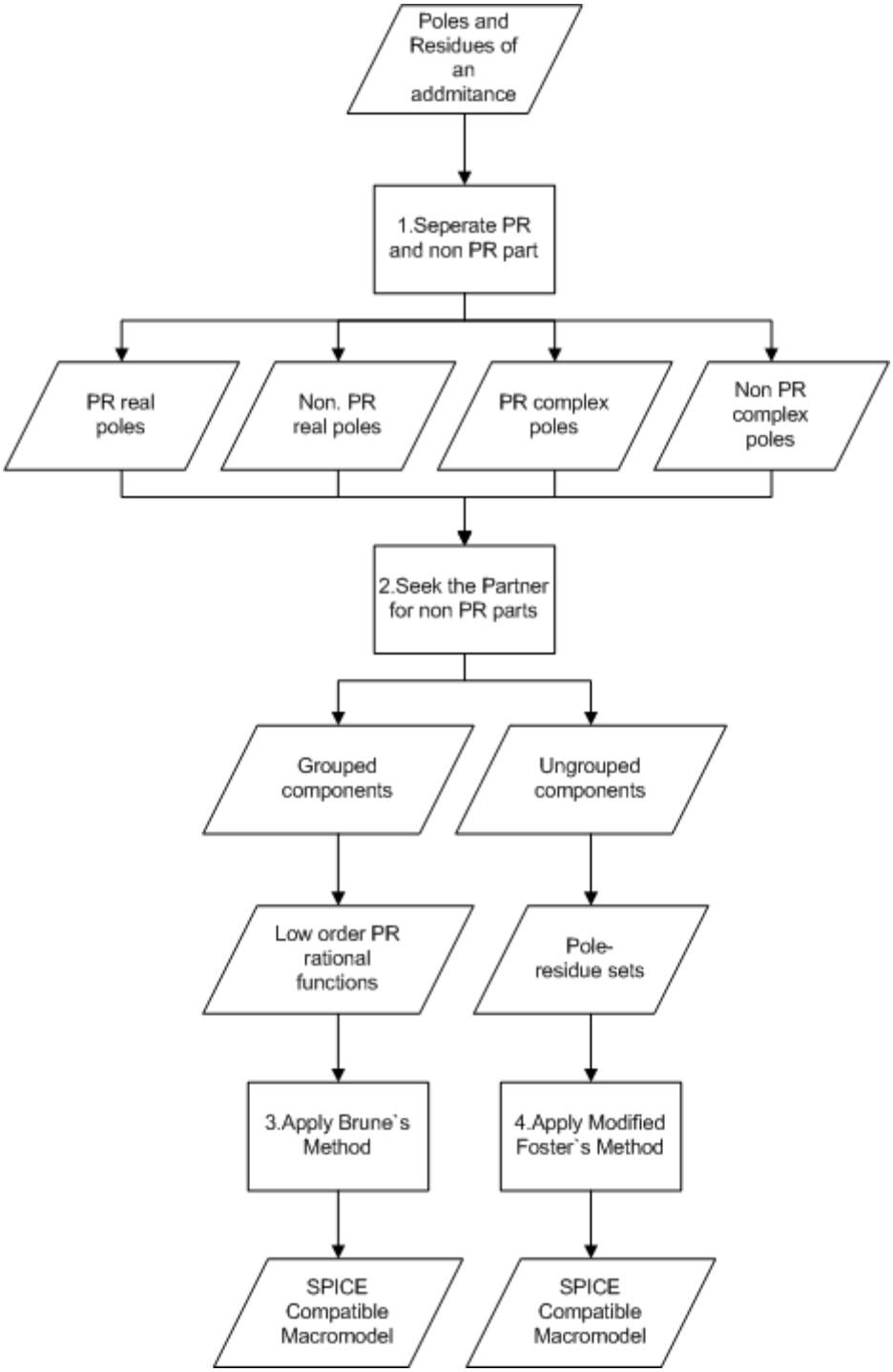


Figure 5.6 Flowchart of Foster-Brune method

For Figure 5.4, it is not difficult to observe that the core part of this algorithm is the second step. Establishing the grouping method efficiently has become the key. A fast grouping algorithm method is given below. Its flowchart is proposed in Figure 5.7.

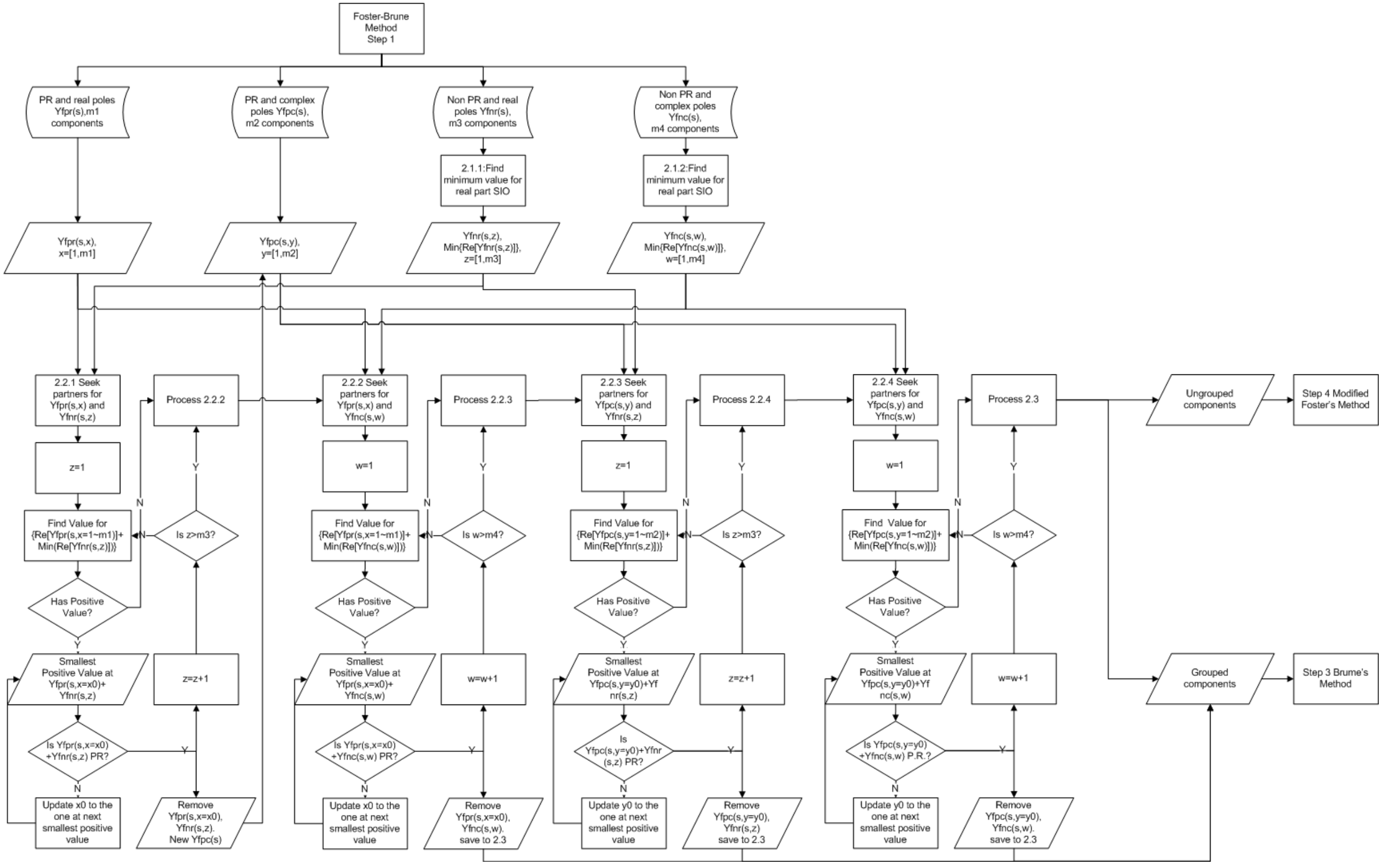


Figure 5.7 Flowchart for fast Grouping Algorithm (step 2) in Foster-Brune method

In real world simulation when number of poles is greater than 20, the results of admittance usually have more non PR components than PR components. Therefore the algorithm seeks the best matching non PR. component for every PR component; it will be more efficient than verse. The most important part for FBM method is in step 2 or the conditions for seeking the partners.

To explain the selection algorithm in Figure 5.6, the selection (or grouping) process is divided into three process 2.1~2.3. From previous step, we separate $y_f(s)$ into four groups: PR components with real poles $y_{fpr}(s)$, PR component with complex poles $y_{fpc}(s)$; Non PR components with real poles $y_{fnr}(s)$ and non PR component with complex poles$y_{fnc}(s)$. When we are search components for grouping $y_{fp}(s)$ and $y_{fn}(s)$, there are four combinations: New components for Brune's process:

- $y_f^{(B1)}(s) = y_{fpr}(s) + y_{fnr}(s) = \frac{c}{s+d} + \frac{a}{s+b}$
- $y_f^{(B2)}(s) = y_{fpr}(s) + y_{fnc}(s) = \frac{c}{s+d} + \frac{a_1 s+a_0}{s^2+b_1 s+b_0}$
- $y_f^{(B3)}(s) = y_{fpc}(s) + y_{fnr}(s) = \frac{c_1 s+c_0}{s^2+d_1 s+d_0} + \frac{a}{s+b}$
- $y_f^{(B4)}(s) = y_{fpc}(s) + y_{fnc}(s) = \frac{c_1 s+c_0}{s^2+d_1 s+d_0} + \frac{a_1 s+a_0}{s^2+b_1 s+b_0}$ (5.16)

were separated and we start from these data.

a. Process 2.1: Find minimum real part of all Non PR components $\min_\omega \text{Re}\left(y_{fn}(j\omega)\right)$. Then sort these components in descending order by value of $\min_\omega \text{Re}\left(y_{fn}(j\omega)\right)$ (**SIDO**).

1) For real poles group $y_{fnr}(s) = \frac{a}{s+b}$; then

$$\text{Re}\left(y_{fnr}(s)\right) = \frac{ab}{\omega^2+b^2},\ \min_\omega \text{Re}\left(y_{fnr}(j\omega)\right) = \text{Re}\left(y_{fnr}(j\omega_0)\right)\Big|_{\omega_0=0} = \frac{a}{b} \quad (5.17)$$

The reorganized data will be:

$$\left\{\left[y_{fnr}^{(1)}(s), \min_\omega \text{Re}\left(y_{fnr}^{(1)}(s)\right), \omega_0^{(1)}\right], \left[y_{fnr}^{(2)}(s), \min_\omega \text{Re}\left(y_{fnr}^{(2)}(s)\right), \omega_0^{(2)}\right], \cdots\right\} \quad (5.18)$$

The first one has the largest value on $\min_\omega \text{Re}\left(y_{fnr}(j\omega)\right)$, or has smallest absolute value. So it has largest possibility to form group with $y_{fp}(s)$.

2) For complex poles group $y_{fnc}(s) = \frac{a_1 s+a_0}{s^2+b_1 s+b_0}$; then

$$\text{Re}\left(y_{fnc}(j\omega)\right) = \text{Re}\left(y_{fnc}(j\Omega)\right)\Big|_{\Omega=\omega^2} = \frac{\Lambda_1 \Omega+\Lambda_0}{\Omega^2+\Gamma_1 \Omega+\Gamma_0} \quad (5.19)$$

where $\Lambda_1 = a_1 b_1 - a_0; \Lambda_0 = a_0 b_0; \Gamma_1 = b_1^2 - 2b_0; \Gamma_0 = b_0^2$

To find $\min_\omega \text{Re}\left(y_{fnc}(j\omega)\right)$, we calculate $\Omega_{1,2}$

$$\Omega_{1,2} = \frac{\Lambda_0 \pm \sqrt{\Lambda_0^2 - \Lambda_0\Lambda_1\Gamma_1 + \Lambda_1^2\Gamma_0}}{\Lambda_1} \; ; \; \Omega_{1,2} \geq 0 \quad (5.20)$$

If Ω_1 or Ω_2 is negative, we remove it since ω is a real number.

Compare three values: $\text{Re}\left(y_{fnc}(j\Omega)\right)\Big|_{\Omega=0} = \frac{a_0}{b_0}$ and $\text{Re}\left(y_{fnc}(j\Omega)\right)\Big|_{\Omega=\Omega_{1,2}}$, the smaller one is $\min_\Omega \text{Re}\left(y_{fn}(j\Omega)\right)$ or $\min_\omega \text{Re}\left(y_{fn}(j\omega)\right)$.

The reorganized data will be:

$$\left\{\left[y_{fnc}^{(1)}(s), \min_\omega \text{Re}\left(y_{fnc}^{(1)}(s)\right), \omega_0^{(1)}\right], \left[y_{fnc}^{(2)}(s), \min_\omega \text{Re}\left(y_{fnc}^{(2)}(s)\right), \omega_0^{(2)}\right], \cdots\right\} \quad (5.21)$$

b. Process 2.2: Find the best potential choice for potential grouping by judging the summation of $\min_\omega \text{Re}\left(y_{fn}(j\omega)\right) + \text{Re}\left(y_{fr}(j\omega)\right)$.

There are 4 possible combinations; their processing order is referred to (5.16). The reason will be explained in Process 5.3. Here we use the first combination $y_{fpr}(s) + y_{fnr}(s)$ as an example.

Suppose $y_{fpr}(s)$ components are $\left[y_{fpr}^{(1)}(s), \cdots, y_{fpr}^{(i)}(s)\right]$, reorganized $y_{fnr}(s)$ components from previous step are $\left[y_{fnr}^{(1)}(s), \cdots, y_{fnr}^{(j)}(s)\right]$.

We start from the first component of $y_{fnr}(s)$, it contains these information:

$$\left[y_{fnr}^{(1)}(s), \min_{\omega=\omega_0^{(1)}} \text{Re}\left(y_{fnr}^{(1)}(s)\right), \omega_0^{(1)}\right] \quad (5.22)$$

From $\omega_0^{(1)}$, we can calculate $\text{Re}\left(y_{fr}(j\omega)\right)$ value for all P.R. components at this frequency and we can form this expression:

$$\left\{\left[\text{Re}\left(y_{fpr}^{(1)}\left(j\omega_0^{(1)}\right)\right) + \text{Re}\left(y_{fnr}^{(1)}\left(j\omega_0^{(1)}\right)\right)\right], \cdots, \left[\text{Re}\left(y_{fpr}^{(i)}\left(j\omega_0^{(1)}\right)\right) + \text{Re}\left(y_{fnr}^{(1)}\left(j\omega_0^{(1)}\right)\right)\right]\right\} \quad (5.23)$$

Suppose **the smallest positive** value happens at kth component of $y_{fpr}(s)$

$$\left[\text{Re}\left(y_{fpr}^{(k)}\left(j\omega_0^{(1)}\right)\right) + \text{Re}\left(y_{fnr}\left(j\omega_0^{(1)}\right)\right)\right] \quad (5.24)$$

then $y_{frk}(s)$ and $y_{fn0}(s)$ are the best potentials for grouping. We need a quick method to prove their sum PR; this method is mention in Appendix IV. If PR condition is not satisfied here, the second smallest positive result in (5.23) become the best potential matching and PR check is applied again until the sum is PR. In most cases, the PR condition can satisfy for first time attempt of this process.

When $y_{fpr}^{(k)}(s)$ and $y_{fnr}^{(1)}$ are form a group, these two components will be removed from original groups and proceed to Process 2.3. Then we seek the partner for the second component of $y_{fnr}(s)$, or $y_{fnr}^{(2)}(s)$.

For lth component $y_{fnr}^{(l)}(s)$, if all values in (5.23) are negatives, then $y_{fnr}^{(l)}(s)$ can't be grouped with any PR component.

Since $\min_\omega \text{Re}\left(y_{fnc}^{(l+1)}(s)\right) \leq \min_\omega \text{Re}\left(y_{fnc}^{(l)}(s)\right)$, the next component $y_{fnr}^{(l+1)}(s)$ has tiny chance to find a partner. We stop this process here and move to next combination.

c. Process 2.3: Collect the components for Brune's method

When PR and non PR component are grouped, they are removed from their original groups and saved in this process for step 3.

There is a special case in Process 2.3. If two components are from $y_{fpr}(s)$ and $y_{fnr}(s)$, their sum is PR and can form a new component in $y_{fpc}(s)$.

$$y_{fpr}(s) + y_{fnr}(s) = \frac{c}{s+d} + \frac{a}{s+b} = \frac{(a+c)s+(ad+bc)}{s^2+(b+d)s+bd} = new\ y_{fpc}(s) \quad (5.25)$$

For this reason, combination for $y_{fpr}(s)$ and $y_{fnr}(s)$ should be firstly executed in Process 2.2 and grouped components are added into $y_{fpc}(s)$ as a new components.

When all four combinations are done in Process 2.2, Process 2.3 will transfer the sum of all saved grouped components into a rational function. It will be the output for Brune's Method in Step 3. The ungrouped components are outputs for Modified Foster's Method in Step 4.

In some cases, there are very few or no PR components in primary data. In these cases, result of Foster-Brune method would be similar to that of Modified Foster's Method. Compare all three methods, the result of Foster-Brune method has less number of components than results of Foster's method and more accurate data than results of Brune's method. Their simulation results will be given in the next chapter to support the analysis.

Chapter 6 Examples and Simulations

In this Chapter, two examples are used for simulation. They are applied VF and LM method to obtain time-domain macromodels at first. The time cost and accuracy are recorded for these simulations. Then Sparse Matrices Algorithm (SMA), Modified Foster's Method (MFM) and Foster-Brune Method (FBM) are applied on them. Number of components and transient analysis are recorded for comparison. To observe their accuracy, we use Backward Euler method on original data for transient analysis.

6.1 2-port transmission line simulation

The first example is a 2-port single transmission line, shown in Figure 6.1. Y-parameter is generated from full-wave simulator. The frequency points for this example are 100 points linearly from 0Hz to 10GHz. When VF method is applied, 20 guessed poles are used and iteration times are maximum 10. The code used for simulation is matrix_fitting _toolbox, which is provided by [74]. The code for LM method is based on [48]. Figure 6.2 and 6.3.also record their results of $|Y_{11}|$ and rms error. Comparison of their data is in Table 6-1.

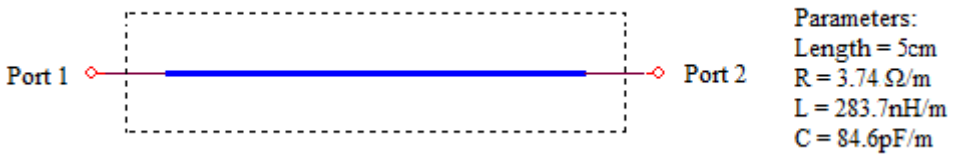


Figure 6.1 Parameters for single transmission line

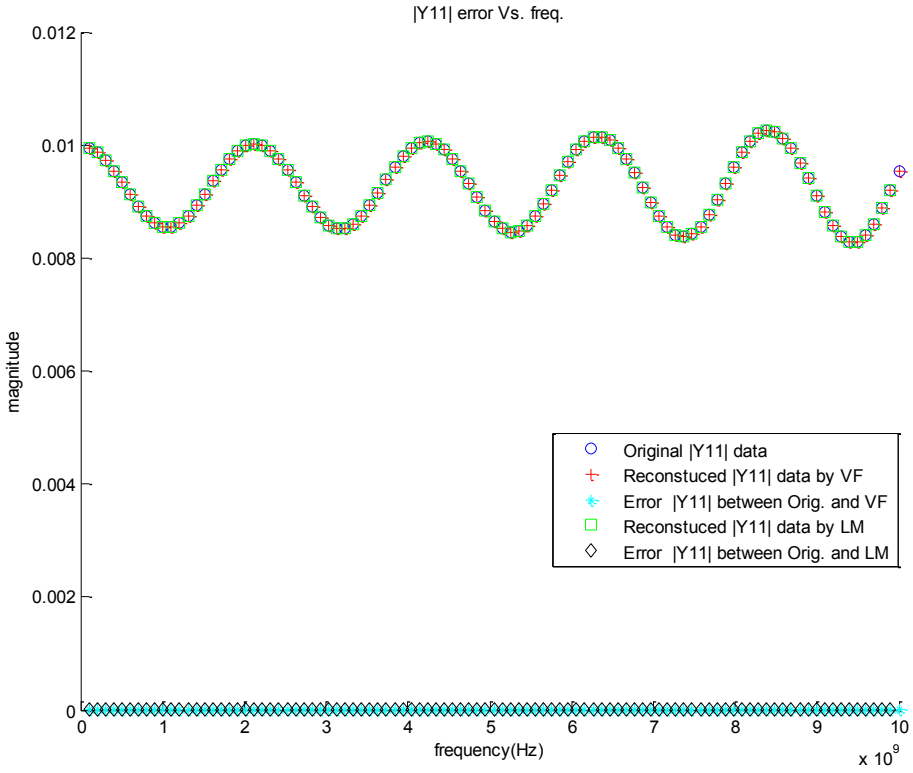


Figure 6.2$|Y_{11}|$ Comparison between Original, VF and LM data on 2-port network

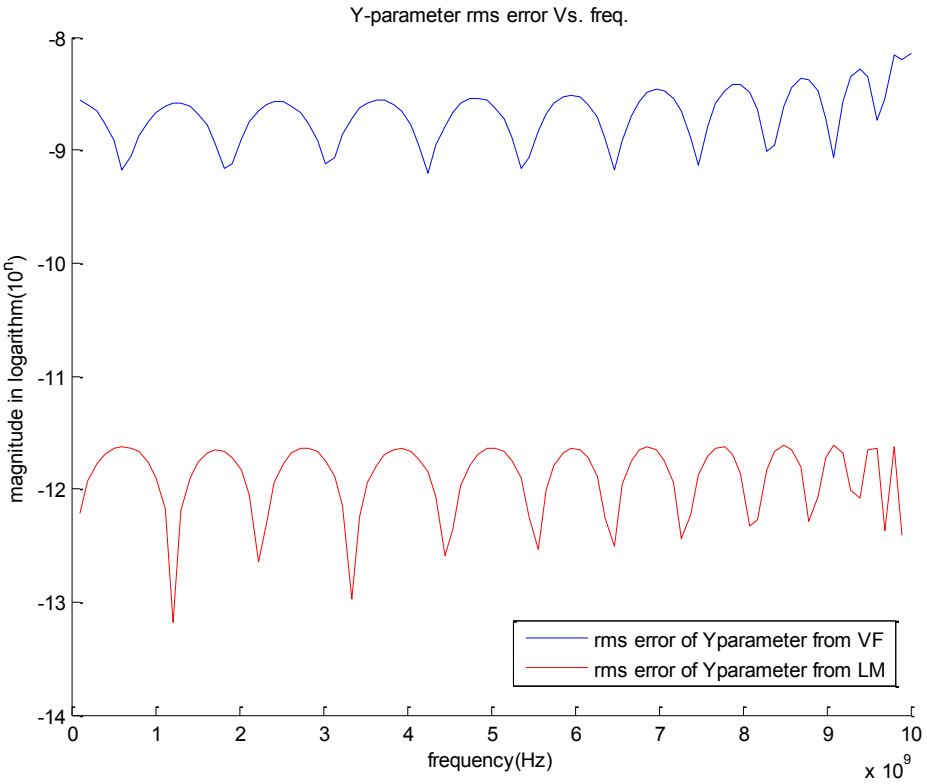


Figure 6.3 Y-paramter rms error Vs. frequency on 2-port network

Table 6-1 Comparison between VF and LM method on 2-port network

	Overall RMS error	Time cost(s)	Order of state space matrices	Number of pole-residues
VF	1.539×10^{-9}	0.1276	40	20
LM	1.629×10^{-12}	0.6105	25	25

After we obtain the time-domain macromodels from VF and LM methods, data are used to generate SPICE netlist through the following method: Sparse Matrices Algorithm (SMA), Modified Foster's method (MFM), Foster and Brune method(FBM). The comparisons of netlists components are stored in table 6-2.

Table 6-2 SPICE netlist generation from different methods for 2-port network

Method	Resistors	Capacitors	Inductors	VCVS	Trans.	Total	Time(s)
SMA	25	36	0	100	0	161	0.0620
MFM	63	49	46	0	0	158	0.0433
FBM	63	49	46	0	0	158	0.0667

From table 6-2, we can figure that components from SMA, MFM and FBM are very close. MFM is faster than SMA. The results of MFM and FBM are same; the reason is that number of non PR components is limited in this simple example, so grouping components are not generated.

To apply the transient analysis to netlist, we suppose a step input voltage ($V_h = 1V, V_l = 0, t_r = 0.1ns, t_s = 10ns, t_r = 0.1ns, delay = 1ns$) connected to port 1 with 50Ω resistor, then connect another 50Ω resistor between port 2 and ground. To test accuracy of each method, Backward Euler method (BEM) [2] is used. BEM uses the global G and C matrices of MNA equation in process of full-wave simulation. Simulation software for SPICE netlist transient analysis is CADENCE Allegro AMS Simulator, and results are shown in Figure 6.4-6.5. Due to FBM and MFM have exactly same results, FBM are not repeated in this example.

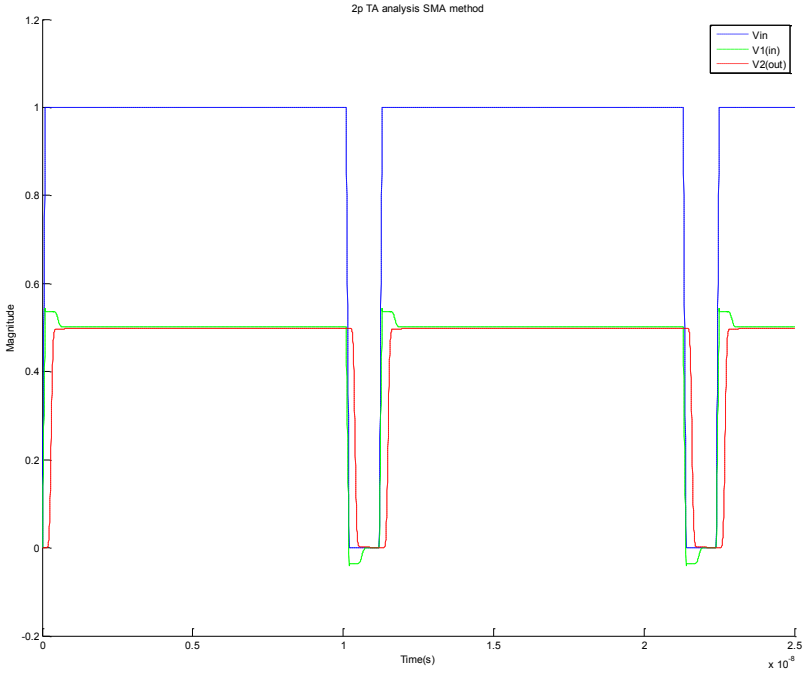


Figure 6.4 Transient Analysis For 2-port network from SMA method

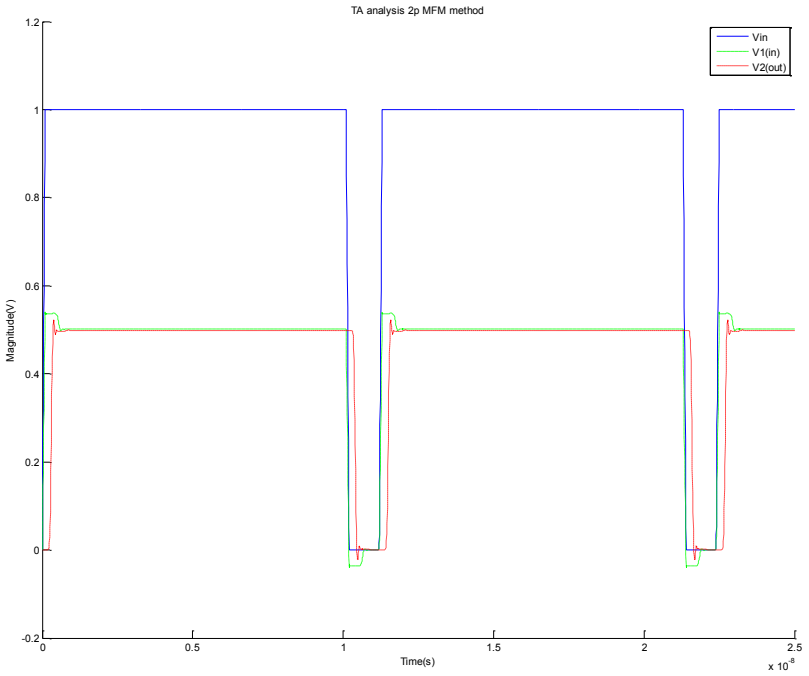


Figure 6.5 Transient Analysis For 2-port network from MFM method

To observe the accuracy of SMA and MFM, we plot the data of output port (port 2) for different methods in Figure 6.6. In Table 6-3, the rms error for SMA and MFM are recorded, compared to results of Backward Euler method. We can comment both methods have excellent accuracy. However, MFM has better accuracy for this example.

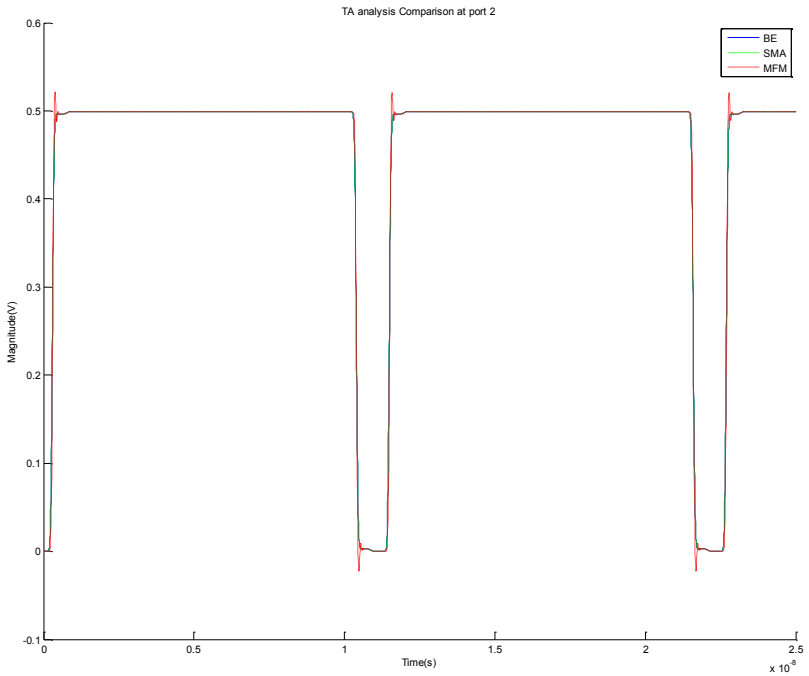


Figure 6.6 Comparison at port 2(output) for 2-port network

Table 6-3 RMS error data for 2-port network

Method	Rms error for V_1	Rms error for V_2
Sparse Matrices Algortihm	1.20×10^{-7}	3.44×10^{-7}
Modified Foster Method	5.28×10^{-6}	8.13×10^{-4}
Foster-Brune Method(same to MFM)	5.28×10^{-6}	8.12×10^{-4}

6.2 8-port network simulation

An 8-port network is simulated in this section,. Its structure and parameters are given in Figure 6.7. Port 1~4 are input ports, port 5~8 are output ports. There are totally 18 transmission lines in this network. Transmission lines T_1~T_9 are coupled transmission line, its parameter RLC per meter will be given in (6.1) in a matrix format. Transmission lines T_{10}~T_{18} are single transmission lines.

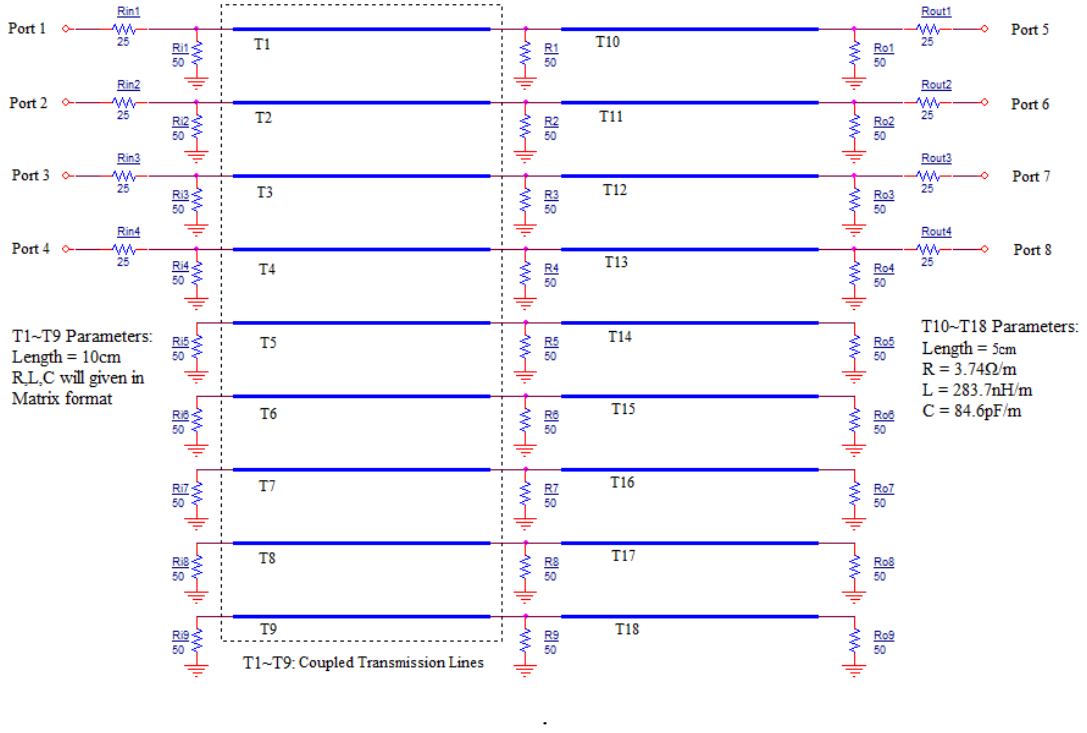


Figure 6.7 Parameters for 8-port network

Parameters for coupled transmission line T_1~T_9: Length = 10cm;

$$C = \begin{bmatrix} 187.6 & -0.3850 & -2.230 & -0.0046 & -2.230 & -0.3850 & -2.230 & -0.0046 & -2.230 \\ -0.3850 & 187.6 & -2.230 & -0.3850 & 0 & 0 & 0 & 0 & -2.230 \\ -2.230 & -2.230 & 187.6 & -2.230 & -0.3850 & 0 & -0.0046 & 0 & -0.3850 \\ -0.0046 & -0.3850 & -2.230 & 187.6 & 0 & 0 & 0 & 0 & 0 \\ -2.230 & 0 & -0.3850 & 0 & 187.6 & -2.230 & -0.3850 & 0 & 0 \\ -0.3850 & 0 & 0 & 0 & -2.230 & 187.6 & -2.230 & -0.3850 & 0 \\ -2.230 & 0 & -0.0046 & 0 & -0.3850 & -2.230 & 187.6 & -2.230 & -0.3850 \\ -0.0046 & 0 & 0 & 0 & 0 & -0.3850 & -2.230 & 187.6 & 0 \\ -2.230 & -2.230 & -0.3850 & 0 & 0 & 0 & -0.3850 & 0 & 187.6 \end{bmatrix} pF/m,$$

$$L = \begin{bmatrix} 504 & 18.2 & 31.5 & 5.70 & 31.5 & 18.2 & 31.5 & 5.70 & 31.5 \\ 18.2 & 504 & 31.5 & 18.2 & 0 & 0 & 0 & 0 & 31.5 \\ 31.5 & 31.5 & 504 & 31.5 & 18.2 & 0 & 5.70 & 0 & 18.2 \\ 5.70 & 18.2 & 31.5 & 504 & 0 & 0 & 0 & 0 & 0 \\ 31.5 & 0 & 18.2 & 0 & 504 & 31.5 & 18.2 & 0 & 0 \\ 18.2 & 0 & 0 & 0 & 31.5 & 504 & 31.5 & 18.2 & 0 \\ 31.5 & 0 & 5.70 & 0 & 18.2 & 31.5 & 504 & 31.5 & 18.2 \\ 5.70 & 0 & 0 & 0 & 0 & 18.2 & 31.5 & 504 & 0 \\ 31.5 & 31.5 & 18.2 & 0 & 0 & 0 & 18.2 & 0 & 504 \end{bmatrix} nH/m, R = \begin{bmatrix} 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 \end{bmatrix} \Omega/m. \quad (6.1)$$

Similarly to previous section, we apply VF and LM method to this example at first. The frequency points are 40 points from 1to 4GHz. For VF method, we again use 20 guessed poles and maximum 10 iteration times. The results of VF and LM method are recorded in Figure 6.8 and 6.9, also in Table 6-4.

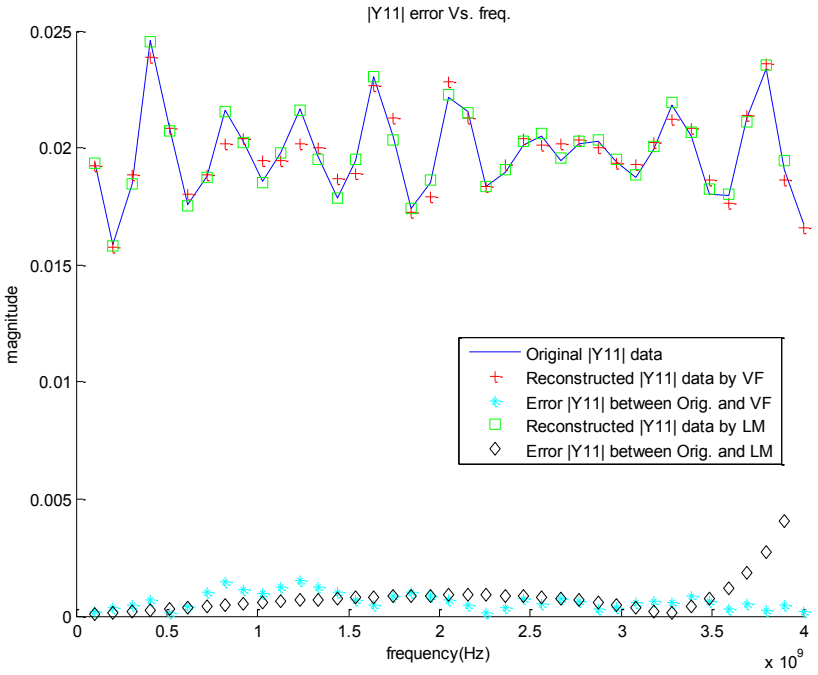


Figure 6.8 $|Y_{11}|$ Comparison between Original, VF and LM data on 8-port network

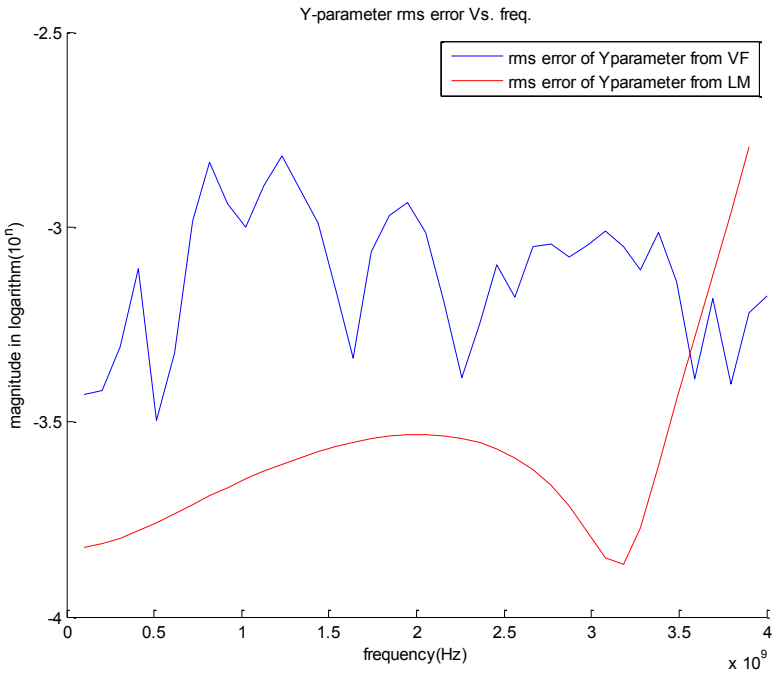


Figure 6.9 Y-paramter rms error Vs. frequency on 8-port network

Table 6-4 Comparison between VF and LM method on 8-port network

	Overall RMS error	Time cost(s)	Order of state space matrices	Number of pole-residues
VF	4.141×10^{-4}	0.3479	160	20
LM	4.1346×10^{-4}	0.7820	44	44

Similar to previous example, these data are used to generate SPICE netlist. The comparisons of netlists components are stored in table 6-5.

Table 6-5 SPICE netlist generation from different methods for 8-port network

Method	Resistors	Capacitors	Inductors	VCVS	Trans.	Total	Time(s)
SMA	80	100	0	704	0	884	0.0882
MFM	756	691	665	0	0	2112	0.2794
FBM	703	620	623	0	24	1970	0.4580

From this result, we can conclude that SMA method is much more robust than Modified Foster's method and Foster-Brune method on dealing with complex systems. It has superior behaviors on both components numbers and time-cost. Compare MFM and FBM, they are a few component groups are generated for Brune's method, thus generate some transformers. However, the optimization number of components is not much ($\approx 10\%$) and time cost is significant. For more complex system, this method would show better performance.

To prove their accuracy, we again apply this simulation like previous example. Simulation results of BEM, SMA, MFM and FBM are plotted in Figure 6.10-6.11 below.

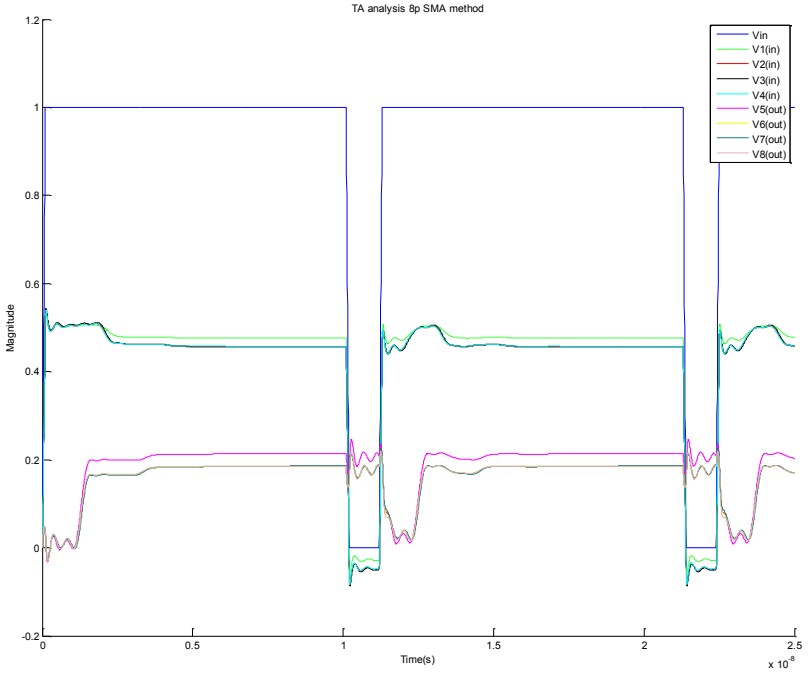


Figure 6.10 Transient Analysis For 8-port network from SMA method

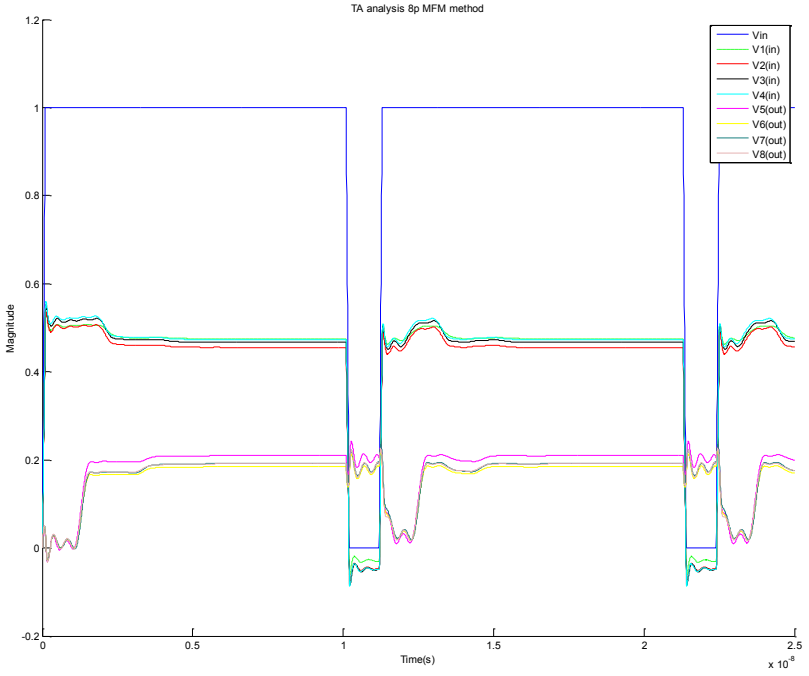


Figure 6.11 Transient Analysis For 8-port network from MFM method

To analyze the accuracy of these methods on this example, we record output data at port 5 from different methods in Figure 6.12 and the rms error for each method in Table 6-6. From the figure, we comment that these data are acceptable. From Table 6-6, we figure that errors of MFM and FBM are very similar since their SPICE netlists don't vary much. Compare these results to that of SMA, they have less errors.

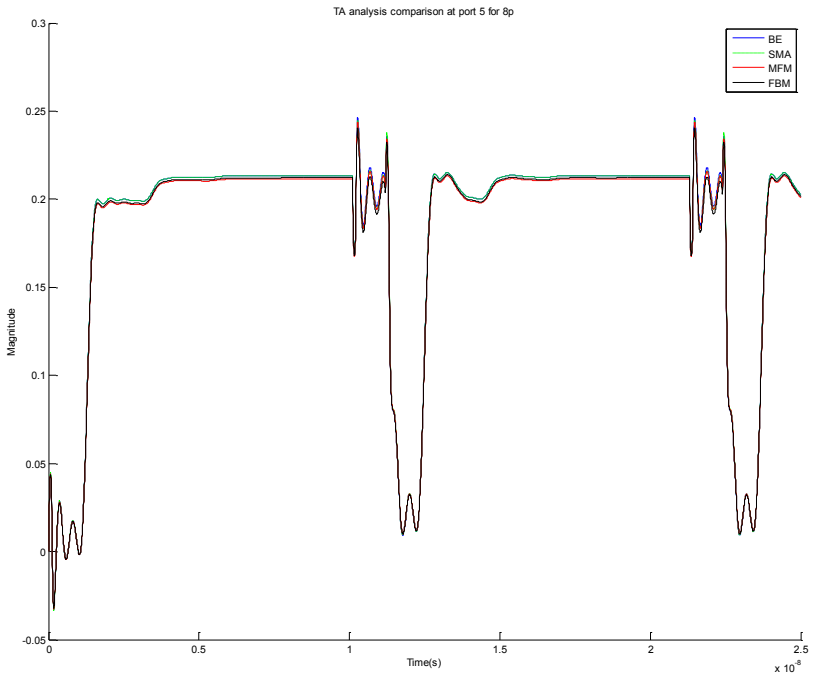


Figure 6.12 Transient Analysis For 8-port network at port 5(output)

Table 6-6 RMS error data for 8-port network

Method	Ave. Rms er. for $V_1 \sim V_4$	Ave. Rms er. for $V_5 \sim V_8$
Sparse Matrices Algorithm	5.22×10^{-4}	2.78×10^{-3}
Modified Foster Method	7.62×10^{-5}	4.28×10^{-4}
Foster-Brune Method(same to MFM)	3.72×10^{-5}	3.73×10^{-4}

From two examples, we can comment on SMA, MFM and FBM. In complicated cases, SMA is superior on netlist generation for less components and fast speed. However, MFM and FBM are robust on handling stability since they generate very limited number of negative components. The results from simulation have proven our hypothesis in Chapter 4 and 5.

Chapter 7 Conclusion and Future Research

7.1 Conclusions

In this thesis, we have described the methods of generating SPICE-compatible macromodels from Y-parameter of unknown systems. These methods are divided into two steps, generating numerical macromodels from Y-parameter and generating SPICE compatible macromodels (SPICE netlist) from numerical macromodels. Chapter 4 focuses on SPICE netlist generation from state space matrices and Chapter 5 focuses on SPICE netlist generation from pole-residue macromodels. There are three important methods proposed in this thesis: **Sparse Matrices Algorithm (SMA)**, **Modified Foster's method (MFM)** and **Foster-Brune method (FBM)**. Compare to other methods in literature survey and background knowledge, these methods have these key advantages:

1. These are all easy to implement. They use the numerical macromodel generated from VF and LM method, which have already had available code. As well, the implementation for Foster's method and Brune's method are not complicated, since they were published a long time ago.

2. All three methods have acceptable accuracy. VF and LM methods are most popular fitting method recently for simulation and they are proven to be accurate.
   In Section 6.1, superiorities of these methods are proven on component numbers and stability.

3. The efficiency of these methods has been proven in the examples. Comparing SMA and MFM, Sparse Matrices Algorithm has fewer components on dealing with complicated network.

4. Both of MFM and FBM can handle various cases in simulation. They are proved to generate minimum negative-value components; although this period is time costy. For simple system macromodeling, MFM are preferred because of time consummation.

5. All these methods have their own advantages; selection through them based on conditions of simulated system would provide better performance.

This thesis is not only focus on how to generate SPICE-compatible marcomodel efficiently, but also provide an overview of all possible methods on macromodeling of unknown system. Section 4.3 provides the sparse model method based on state-space matrices of vector fitting results, although this method has some drawbacks on massive number of components. When the idea of Figure 1.1 is established; all possible methods are acceptable for a trial, although some of them could not give satisfied simulation results. To qualify the final results (SPICE netlist), there parameters can be taken into account: Accuracy of macromeodel (netlist), complexity (number of the components) and stability (minimize negative-value and imaginary components). For different inputting system, the most efficient method might be different. Within the creation of new fitting methods, more algorithms will be invented based their characteristics. Maintaining an open-mind on all possible simulation method is another main idea of this thesis.

7.2 Future Research

As it mentioned above, this macromodeling problem does not have the only one solution. The further researches of these methods include three parts: optimizing the existing method, seeking new possible algorithms and alternating frequency-domain data.

We also can apply sparse model algorithm on VF method; however, the result is not always stable when port number of unknown system is large (>20). Due to these reasons, we skip this method in simulation. Possible optimizations can be made to Sparse Model Method to adapt the state-space matrices from vector fitting method. Moreover, for complicated systems(over 100ports), both passivity violation and stability violation sometime occur. Further optimizations on all proposed methods may further increase the accuracy and stability on large-system simulations.

The second part involves other fitting methods, such as those mentioned in literature survey or the new methods in the future. Based on the properties of these fitting methods, new algorithms can be investigated.

The last part is to derive SPICE netlist from frequency-domain data in other formats. For instance, S-parameter is an alternative choice. The methods to generate SPICE netlist from S-parameter are very different to those discussed in this thesis. It would be another topic.

APPENDX I Transformation between state space matrices and rational approximation

This section would introduce the transformation methods between two model patterns: state space matrices and pole-residue approximation. Before we start the algorithms, it is very necessary to introduce these two models:

1. State space matrices model

   State space time-domain model has similar formulation pattern to MNA equation; however, this model describes the port relationship. Its form for the black-box system in Figure 2.3 is given in (2.9).

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t) + Du(t) \qquad (1)$$

   where

   - $u(t)$ is the vector of port voltages, p is the port number.

$$u(t) = [v_1(t) \quad v_2(t) \quad \cdots \quad v_p(t)]^T. \ (2)$$

   - $y(t)$ is the vector of port current:

$$y(t) = [i_1(t) \quad i_2(t) \quad \cdots \quad i_p(t)]^T. \ (3)$$

   - $x(t)$ is the vector of internal statement of this system, q is the number of statement

$$x(t) = [x_1(t) \quad x_2(t) \quad \cdots \quad x_q(t)]^T \ . \ (4)$$

   Matrices A, B, C and D are called state-space variable and their sizes are:
   $A \in \mathbb{R}^{q\times q}, B \in \mathbb{R}^{q\times p}, C \in \mathbb{R}^{p\times q}, D \in \mathbb{R}^{p\times p}$ and q is usually larger than p.
   In frequency domain, (2.9) will be:

$$sX = AX + Bu$$
$$y = CX + Du \qquad (5)$$

   Y matrix can be solved through flowing algorithm:
   Solve X in first equation of (2.18):

$$X = (sI - A)^{-1}Bu \qquad (6)$$

   Then substitute (2.19) into the second equation of (2.18)

$$y = [C(sI - A)^{-1}B + D]u \qquad (7)$$

   Since y are port currents and u are port voltages, (2.20) transforms into

$$Y = C(sI - A)^{-1}B + D \qquad (8)$$

In some algorithms [23, 37], the results have state space matrix D equals to 0.
For better fitting in Loewner Matrix Method, term proportional to s can be included in transfer function. [47]

Matrix Y^{∞} is added into time-domain model and (1) becomes

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t) + Du(t) + Y^{\infty}\dot{u}(t) \qquad (9)$$

Where Y^{∞} is a positive real matrix with size equals to Y [47],

$$Y = C(sI - A)^{-1}B + D + sY^{\infty} \qquad (10)$$

There are a few ways to extract state space model in (7) or (10), such as Asymptotic Waveform Evaluation and Loewner Matrix Method.

2. Pole-residue Rational Approximation Model

This model uses the concept of Pade approximation to approximate our desired data [17], the transfer function H for a response can be expressed with number of its moments:

$$H(s) = m_0 + m_1 s + m_2 s^2 + \cdots \qquad (11)$$

Pade approximation fits a rational approximate transfer function to $H(s)$:

$$H(s) = \frac{a_0 + a_1 s + a_2 s^2 + \cdots + a_L s^L}{1 + b_1 s + b_2 s^2 + \cdots + b_M s^M};\ a, b \in \mathbb{R} \quad (12)$$

Where L/M equals to the order of (s) .
For Y-parameter, the approximation of $Y_{ij}(s)$ can be defined as a numerator $N(s)$ divided by a denominator $D(s)$ with a same order.

$$Y_{ij}(s) = \frac{N(s)}{D(s)} = \frac{a_0 + a_1 s + a_2 s^2 + \cdots + a_N s^N}{b_0 + b_1 s + b_2 s^2 + \cdots + b_N s^N};\ a, b \in \mathbb{R} \quad (13)$$

Equation (13) states the rational function format of the approximation; several other formats can be derived from this function:
Pole-residue format:

$$Y_{ij}(s) = \frac{a_0 + a_1 s + a_2 s^2 + \cdots + a_N s^N}{b_0 + b_1 s + b_2 s^2 + \cdots + b_N s^N} = \sum_{n=1}^{N} \frac{r_n}{s - p_n} + d + sh; r, p, d, h \in \mathbb{C} \qquad (14)$$

Pole-zeros format:

$$Y_{ij}(s) = \frac{a_0 + a_1 s + a_2 s^2 + \cdots + a_N s^N}{b_0 + b_1 s + b_2 s^2 + \cdots + b_N s^N} = h\frac{\prod_{n=1}^{N}(s - z_n)}{\prod_{n=1}^{N}(s - p_n)}; z, p, h \in \mathbb{C} \qquad (15)$$

Now we can introduce the transformation algorithms between these two models. At first, we suppose a P-port system has Y-parameter $Y(s)$:

$$I(s) = Y(s)V(s) \Leftrightarrow \begin{bmatrix} I_1(s) \\ \vdots \\ I_2(s) \end{bmatrix} = \begin{bmatrix} Y_{11}(s) & \cdots & Y_{1P}(s) \\ \vdots & \ddots & \vdots \\ Y_{P1}(s) & \cdots & Y_{PP}(s) \end{bmatrix} \begin{bmatrix} V_1(s) \\ \vdots \\ V_2(s) \end{bmatrix} \quad (16)$$

If we apply VF method (Section 3.3.2) on $Y(s)$, and N guessed poles are used; then the result would be sets of pole-reisdues parameters:

$$\left\{ p_n, r_n^{(ij)} \in \mathbb{C}; d^{(ij)}, h^{(ij)} \in \mathbb{R} \right\}; n \in [1, N]; i, j \in [1, P] \text{ and}$$

$$\begin{bmatrix} Y_{11}(s) & \cdots & Y_{1P}(s) \\ \vdots & \ddots & \vdots \\ Y_{P1}(s) & \cdots & Y_{PP}(s) \end{bmatrix} = \begin{bmatrix} \left[\sum_{n=1}^{N} \frac{r_n^{(11)}}{s-p_n} + d^{(11)} + sh^{(11)} \right] & \cdots & \left[\sum_{n=1}^{N} \frac{r_n^{(1P)}}{s-p_n} + d^{(1P)} + sh^{(1P)} \right] \\ \vdots & \ddots & \vdots \\ \left[\sum_{n=1}^{N} \frac{r_n^{(P1)}}{s-p_n} + d^{(P1)} + sh^{(P1)} \right] & \cdots & \left[\sum_{n=1}^{N} \frac{r_n^{(PP)}}{s-p_n} + d^{(PP)} + sh^{(PP)} \right] \end{bmatrix} \quad (17)$$

If we apply LM method (Section 3.4.2) on $Y(s)$ and q internal statement is generated through the process; then the results would be several state space matrices:

$$\{A, B, C, D, E, Y^{\infty}\}; A, E \in \mathbb{R}^{q \times q}; B \in \mathbb{R}^{q \times P}; C \in \mathbb{R}^{P \times q}; D, Y^{\infty} \in \mathbb{R}^{P \times P};$$

$$sEx = Ax + Bu; y = Cx + Du + sY^{\infty}u;$$

$$Y(s) = \begin{bmatrix} Y_{11}(s) & \cdots & Y_{1P}(s) \\ \vdots & \ddots & \vdots \\ Y_{P1}(s) & \cdots & Y_{PP}(s) \end{bmatrix} = C(sE - A)^{-1}B + D + sY^{\infty} \quad (18)$$

In later part, algorithms are given for transformation between $\left\{ p_n, r_n^{(ij)}, d^{(ij)}, h^{(ij)} \right\}$ and $\{A, B, C, D, E, Y^{\infty}\}$. In [23], some clues are given for state space realization; and more details would be given in this section.

A. Transformation from pole-residue model to state space matrices model

At first we compress the $Y(s)$ of (17) and (18):

$$\begin{bmatrix} \left[\sum_{n=1}^{N} \frac{r_n^{(11)}}{s-p_n} + d^{(11)} + sh^{(11)} \right] & \cdots & \left[\sum_{n=1}^{N} \frac{r_n^{(1P)}}{s-p_n} + d^{(1P)} + sh^{(1P)} \right] \\ \vdots & \ddots & \vdots \\ \left[\sum_{n=1}^{N} \frac{r_n^{(P1)}}{s-p_n} + d^{(P1)} + sh^{(P1)} \right] & \cdots & \left[\sum_{n=1}^{N} \frac{r_n^{(PP)}}{s-p_n} + d^{(PP)} + sh^{(PP)} \right] \end{bmatrix} = C(sE - A)^{-1}B + D + sY^{\infty} \quad (19)$$

The next step to extract d and h part on the left side:

$$\begin{bmatrix} \sum_{n=1}^{N} \frac{r_n^{(11)}}{s-p_n} & \cdots & \sum_{n=1}^{N} \frac{r_n^{(1P)}}{s-p_n} \\ \vdots & \ddots & \vdots \\ \sum_{n=1}^{N} \frac{r_n^{(P1)}}{s-p_n} & \cdots & \sum_{n=1}^{N} \frac{r_n^{(PP)}}{s-p_n} \end{bmatrix} + \begin{bmatrix} d^{(11)} & \cdots & d^{(1P)} \\ \vdots & \ddots & \vdots \\ d^{(P1)} & \cdots & d^{(PP)} \end{bmatrix} + s \begin{bmatrix} h^{(11)} & \cdots & h^{(1P)} \\ \vdots & \ddots & \vdots \\ h^{(P1)} & \cdots & h^{(PP)} \end{bmatrix} = C(sE - A)^{-1}B + D + sY^{\infty} \quad (20)$$

D and Y^{∞} can be easily identified:

$$D = \begin{bmatrix} d^{(11)} & \cdots & d^{(1P)} \\ \vdots & \ddots & \vdots \\ d^{(P1)} & \cdots & d^{(PP)} \end{bmatrix} \text{and } Y^{\infty} = \begin{bmatrix} h^{(11)} & \cdots & h^{(1P)} \\ \vdots & \ddots & \vdots \\ h^{(P1)} & \cdots & h^{(PP)} \end{bmatrix} (21)$$

The rest part leaves:

$$C(sE - A)^{-1}B = \begin{bmatrix} \sum_{n=1}^{N} \frac{r_n^{(11)}}{s-p_n} & \cdots & \sum_{n=1}^{N} \frac{r_n^{(1P)}}{s-p_n} \\ \vdots & \ddots & \vdots \\ \sum_{n=1}^{N} \frac{r_n^{(P1)}}{s-p_n} & \cdots & \sum_{n=1}^{N} \frac{r_n^{(PP)}}{s-p_n} \end{bmatrix} (22)$$

To derive A, B, C and E matrices, definition of Jordan-canonical[68] is applied,

$$A = \begin{bmatrix} p_1 \boldsymbol{I_P} & 0 & \cdots & 0 \\ 0 & p_2 \boldsymbol{I_P} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & p_N \boldsymbol{I_P} \end{bmatrix}, A \in \mathbb{C}^{NP \times NP};$$

$$B = [\boldsymbol{I_P} \quad \boldsymbol{I_P} \quad \cdots \quad \boldsymbol{I_P}]^T, B \in \mathbb{R}^{NP \times P}$$

$$C = \begin{bmatrix} r_1^{(11)} & \cdots & r_1^{(1P)} & \cdots & r_N^{(11)} & \cdots & r_N^{(1P)} \\ \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ r_1^{(P1)} & \cdots & r_1^{(PP)} & \cdots & r_N^{(P1)} & \cdots & r_N^{(PP)} \end{bmatrix} = [\boldsymbol{r_1} \quad \boldsymbol{r_2} \quad \cdots \quad \boldsymbol{r_N}], C \in \mathbb{C}^{P \times NP}$$

$$E = \boldsymbol{I_{NP}} \quad (23)$$

Where $\boldsymbol{I_P}$ is an identity matrix with size $P \times P$. E is an identity matrix with size $NP \times NP$.

In this model, A and C are complex matrices; a transformation matrix $\boldsymbol{T}$ is introduced to transform them into real matrices:

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{I_P} & \boldsymbol{I_P} \\ i\boldsymbol{I_P} & -i\boldsymbol{I_P} \end{bmatrix}, \boldsymbol{T}^{-1} = \begin{bmatrix} \frac{1}{2}\boldsymbol{I_P} & \frac{-i}{2}\boldsymbol{I_P} \\ \frac{1}{2}\boldsymbol{I_P} & \frac{i}{2}\boldsymbol{I_P} \end{bmatrix} (24)$$

Suppose p_i and p_{i+1} are complex conjugate pole pairs; p_j is a real pole, construct overall transformation matrix $\widehat{\boldsymbol{T}}$ as

$$A = \begin{bmatrix} \ddots & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & p_j \boldsymbol{I_P} & 0 & \cdots & \cdots & 0 \\ \vdots & 0 & \ddots & 0 & \cdots & 0 \\ \vdots & \vdots & 0 & p_i \boldsymbol{I_P} & 0 & 0 \\ \vdots & \vdots & \vdots & 0 & p_{i+1} \boldsymbol{I_P} & 0 \\ 0 & 0 & 0 & 0 & 0 & \ddots \end{bmatrix}, \widehat{\boldsymbol{T}} = \begin{bmatrix} \ddots & 0 & \cdots & \cdots & 0 \\ 0 & \boldsymbol{I_P} & 0 & \cdots & 0 \\ \vdots & 0 & \ddots & 0 & 0 \\ \vdots & \vdots & 0 & \boldsymbol{T} & 0 \\ 0 & 0 & 0 & 0 & \ddots \end{bmatrix} (25)$$

Use the equations in (3) to apply realization transformation, E is omitted because of identity matrix. Suppose $\hat{x} = \widehat{\boldsymbol{T}}x$:

$$sx = Ax + Bu \Longrightarrow s\hat{x} = \widehat{\boldsymbol{T}}A\widehat{\boldsymbol{T}}^{-1}\hat{x} + \widehat{\boldsymbol{T}}Bu \Leftrightarrow s\hat{x} = \hat{A}\hat{x} + \hat{B}u$$

$$y = Cx + Du + sY^{\infty}u \Longrightarrow y = \hat{C}\hat{x} + Du + sY^{\infty}u \text{ (26)}$$

In new model, $\hat{A}$, $\hat{B}$ and $\hat{C}$ are real matrices.

$$\hat{A} = \widehat{\boldsymbol{T}}A\widehat{\boldsymbol{T}}^{-1} = \begin{bmatrix} \ddots & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & p_j\boldsymbol{I_P} & 0 & \cdots & \cdots & 0 \\ \vdots & 0 & \ddots & 0 & \cdots & 0 \\ \vdots & \vdots & 0 & Re(p_i)\boldsymbol{I_P} & Im(p_i)\boldsymbol{I_P} & 0 \\ \vdots & \vdots & \vdots & -Im(p_i)\boldsymbol{I_P} & Re(p_i)\boldsymbol{I_P} & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \ddots \end{bmatrix}, A \in \mathbb{R}^{NP\times NP};$$

$$\hat{B} = \widehat{\boldsymbol{T}}B = [\cdots \quad \boldsymbol{I_P} \quad \cdots \quad 2\boldsymbol{I_P} \quad 0 \quad \cdots]^T, B \in \mathbb{R}^{NP\times P};$$

$$\hat{C} = C\widehat{\boldsymbol{T}}^{-1} = [\cdots \quad \boldsymbol{r_j} \quad \cdots \quad Re(\boldsymbol{r_i}) \quad Im(\boldsymbol{r_i}) \quad \cdots], C \in \mathbb{R}^{P\times NP} \text{ (27)}$$

Therefore, state space matrices transformed from $\left\{p_n, r_n^{(ij)}, d^{(ij)}, h^{(ij)}\right\}$ are $\left\{\hat{A}, \hat{B}, \hat{C}, D, E, Y^{\infty}\right\}$, their expressions are in (21) and (27).

B. Transformation from state space matrices model to pole-residue model

Start from the state space matrices equation:

$$sEx = Ax + Bu; y = Cx + Du + sY^{\infty}u. \text{ (28)}$$

Rational approximation for P ports system

$$H(s) \cong Y(s) \cong \frac{y}{u}, H(s) \in \mathbb{C}^{P\times P} \text{ (29)}$$

To obtain $H(s)$, we have to modify (28). Since E is often close to singular[55], we multiply A^{-1} on both sides of the first equation of (28), then it becomes

$$sA^{-1}Ex = x + A^{-1}Bu; y = Cx + Du + sY^{\infty}u. \text{ (30)}$$

Then find eigenvectors and eigenvalues for $A^{-1}E$, assume that

$$V^{-1}(A^{-1}E)V = \Gamma; V, \Gamma \in \mathbb{R}^{q\times q} \text{ (31)}$$

Where V is eigenvectors and Γ is square matrix where eigenvalues are on diagonal entry. Suppose $x = V\tilde{x}$, then (30) is transformed into

$$s(A^{-1}E)V\tilde{x} = V\tilde{x} + A^{-1}Bu \Leftrightarrow s\Gamma\tilde{x} = \tilde{x} + V^{-1}A^{-1}Bu$$

$$\Leftrightarrow \tilde{x} = (s\Gamma - I)^{-1}V^{-1}A^{-1}Bu \text{ ;}$$

$$y = CV\tilde{x} + Du + sY^{\infty}u. \text{ (32)}$$

In this case, substitute $\tilde{x}$ into y and obtain $H(s)$:

$$H(s) = \frac{y}{u} = CV(s\Gamma - I)^{-1}V^{-1}A^{-1}B + D + sY^{\infty} = \tilde{C}(s\Gamma - I)^{-1}\tilde{B} + D + sY^{\infty};$$

$$\tilde{C} = CV; \tilde{B} = V^{-1}A^{-1}B; \ \tilde{C} \in \mathbb{C}^{P\times q}; \tilde{B} \in \mathbb{C}^{q\times P} \text{ (33)}$$

From the definition of eigenvalues, $\Gamma = diag(\lambda_1, \lambda_2, \cdots, \lambda_q)$. To calculate poles and residues sets, we suppose a set of parameter Λ that:

$$\Lambda = (s\Gamma - I)^{-1} = diag(\Lambda_1, \Lambda_2, \cdots, \Lambda_q); \Lambda_i = \frac{1/\lambda_i}{s-1/\lambda_i}, i \in [1, q] \quad (34)$$

Then substitute Λ into (33)

$$H(s) = \tilde{C}\Lambda\tilde{B} + D + sY^{\infty} = \begin{bmatrix} \tilde{c}_{11} & \cdots & \tilde{c}_{1q} \\ \vdots & \ddots & \vdots \\ \tilde{c}_{P1} & \cdots & \tilde{c}_{Pq} \end{bmatrix} \begin{bmatrix} \Lambda_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \Lambda_q \end{bmatrix} \begin{bmatrix} \tilde{b}_{11} & \cdots & \tilde{b}_{1P} \\ \vdots & \ddots & \vdots \\ \tilde{b}_{q1} & \cdots & \tilde{b}_{qP} \end{bmatrix} + \begin{bmatrix} d_{11} & \cdots & d_{1P} \\ \vdots & \ddots & \vdots \\ d_{P1} & \cdots & d_{PP} \end{bmatrix} + s \begin{bmatrix} y_{11}^{\infty} & \cdots & y_{1P}^{\infty} \\ \vdots & \ddots & \vdots \\ y_{P1}^{\infty} & \cdots & y_{PP}^{\infty} \end{bmatrix} \quad (35)$$

Since $\tilde{C}, \tilde{B}$ are full matrices, expand $H(s)$ into individual component:

$$H_{ij} = \frac{\tilde{c}_{i1}\tilde{b}_{1j}/\lambda_1}{s-1/\lambda_1} + \frac{\tilde{c}_{i2}\tilde{b}_{2j}/\lambda_2}{s-1/\lambda_2} + \cdots + \frac{\tilde{c}_{iq}\tilde{b}_{qj}/\lambda_q}{s-1/\lambda_q} + d_{ij} + sy_{ij}^{\infty}, i \in [1, P], j \in [1, P] \quad (36)$$

Now all elements in H has same expression format to the rational approximation of Y-parameter. Compare (36) to equation (17), we can obtain:

$$H_{ij}(s) = \frac{r_1{}^{(ij)}}{s - p_1} + \frac{r_2{}^{(ij)}}{s - p_2} + \cdots + \frac{r_q{}^{(ij)}}{s - p_q} + d^{(ij)} + sh^{(ij)};$$

$$p_n = \frac{1}{\lambda_n}, r_n{}^{(ij)} = \frac{\tilde{c}_{in}\tilde{b}_{nj}}{\lambda_n}, d^{(ij)} = d_{ij}, h^{(ij)} = y_{ij}^{\infty}; i, j \in [1, P]; n \in [1, q] \quad (37)$$

Expression for $\left\{p_n, r_n^{(ij)}, d^{(ij)}, h^{(ij)}\right\}$ are all given in (37); and pole sets are $(1/\lambda_1, 1/\lambda_2, \cdots, 1/\lambda_q)$. Since all eigenvalues of $A^{-1}E$ are negative on its real part; all poles are on the left side of complex-real plane, in another word, they are all stable.

APPENDX II All the cases for Modified Foster Method

To show all the cases of Modified Foster Method, we start from the admittance on 1-port system of n poles-residue format

$$y(s) = \frac{r_1}{s-p_1} + \frac{r_2}{s-p_2} + \cdots + \frac{r_{n-1}}{s-p_{n-1}} + \frac{r_n}{s-p_n} + d + hs \qquad (1)$$

For every step, **MFM** considers $y_f(s)$, and write the components of $y_f(s)$ to the netlist.

$$y(s) = \frac{r_x}{s-p_x} + y'(s) = y_f(s) + y'(s) \quad (2)$$

If p_x is real, then

$$y_f(s) = \frac{r_x}{s-p_x};\ p_x, r_x \in \boldsymbol{R}, r_x \neq 0, p_x \leq 0. \ (3)$$

Assign $a = r, b = -p; a, b \in \boldsymbol{R}, a \neq 0, b \geq 0$, Then

$$y_f(s) = \frac{a}{s+b} \ (4)$$

There are three cases for p_x is a real pole, one PR case ($y_f(s)$ is PR), one non PR case and one limit case (some value is 0).

1. Case PR: $a > 0, b > 0$

$$y_f(s) = \frac{a}{s+b} = \frac{1}{\frac{s}{a}+\frac{b}{a}} = \frac{1}{Ls+R},$$

$$L = \frac{1}{a}, R = \frac{b}{a}. \ (5)$$

2. Case non PR: $a < 0, b > 0$

   In this case, negative values will be generated if we follow method in previous case. The solution is to add a positive admittance G_0 to $y_f(s)$. G_0 will be absorbed by $y(s)$, or the value of d in (1) changes to

$$d' = d - G_0. \ (6)$$

   then

$$y'_f(s) = y_f(s) + G_0 = \frac{a}{s+b} + G_0 = \frac{G_0 s + (bG_0 + a)}{s+b} \ (7)$$

   Let $G_0 = -\frac{a}{b}; G_0 > 0$

$$y'_f(s) = \frac{G_0 s}{s+b} = \frac{1}{\frac{1}{G_0}+\frac{b}{G_0 s}} = \frac{1}{R+\frac{C}{s}},$$

$$C = \frac{G_0}{b} = -\frac{a}{b^2}, R = -\frac{b}{a} \ (8)$$

3. Case Limit: $b = 0\ /p = 0$

   In this case, negative values may occur when a<0. However, limit case happens rarely in real world simulation. So

$$y_f(s) = \frac{a}{s}, L = \frac{1}{a} \quad (9)$$

If p_x is not real, then

$$y_f(s) = \frac{r_x}{s-p_x} + \frac{r_{x+1}}{s-p_{x+1}} = \frac{r_x}{s-p_x} + \frac{r_x^*}{s-p_x^*} \quad (10)$$

Suppose

$$r_x = \alpha + j\beta, p_x = -\sigma + j\theta; (\alpha, \beta, \sigma, \theta)\epsilon \boldsymbol{R}, \sigma \geq 0, \theta > 0, (\alpha \neq 0) \cup (\beta \neq 0) \quad (11)$$

Then

$$y_f(s) = \frac{a_1 s + a_0}{s^2 + b_1 s + b_0} \quad (12)$$

Where

$$a_1 = 2\alpha, a_0 = 2(\alpha\sigma - \beta\theta), b_1 = 2\sigma, b_0 = \sigma^2 + \theta^2 \quad (13)$$

From (13), we also can infer that

$$b_0 > 0, b_1 \geq 0, (a_1, a_0, b_1, b_0)\epsilon \boldsymbol{R} \quad (14)$$

In the first chapter of [56], the PR condition for (12) is given:

$$a_0 b_0 + (a_1 b_1 - a_0)\omega^2 \geq 0 \text{ or } a_1 b_1 \geq a_0 \geq 0 \text{ or } \frac{\alpha}{|\beta|} \geq \frac{\theta}{\sigma} \quad (15)$$

There are totally seven cases for p_x is not a real pole, one PR case ($y_f(s)$ is PR), two non PR cases and four limit cases (some value is 0). The possible limit cases happens when $b_1 = 0$ or $a_1 = 0$ or $a_0 = 0$, a_1 and a_0 cannot be zero at the same time. However, limit cases happens very rare in real world simulation that they almost can be neglect. In PR case and non PR cases, we assume $b_0 > 0, b_1 > 0, a_1 \neq 0, a_0 \neq 0$.

1. Case PR: $a_1 b_1 \geq a_0 > 0$

$$z_f(s) = \frac{1}{y_f(s)} = \frac{s^2 + b_1 s + b_0}{a_1 s + a_0} = \frac{s}{a_1} + \frac{\left(b_1 - \frac{a_0}{a_1}\right)s + b_0}{a_1 s + a_0} = \frac{s}{a_1} + \frac{\left(b_1 - \frac{a_0}{a_1}\right)}{a_1} + \frac{b_0 - \left(b_1 - \frac{a_0}{a_1}\right)\frac{a_0}{a_1}}{a_1 s + a_0}$$

$$= \frac{s}{a_1} + \left(\frac{a_1 b_1 - a_0}{{a_1}^2}\right) + \frac{b_0 - a_0\left(\frac{a_1 b_1 - a_0}{{a_1}^2}\right)}{a_1 s + a_0} = Ls + R + \frac{1}{C_p s + 1/R_p} \quad (16)$$

   So the components are:

$$L = \frac{1}{a_1}, R = \frac{a_1 b_1 - a_0}{{a_1}^2}, C_p = \frac{{a_1}^3}{{a_1}^2 b_0 - a_0 a_1 b_1 + {a_0}^2}, R_p = \frac{{a_1}^2 b_0 - a_0 a_1 b_1 + {a_0}^2}{a_0 {a_1}^2} \quad (17)$$

To prove all these values are non-negative:

Form $a_1 b_1 \geq a_0 > 0$ and $b_1 > 0$, we obtain $a_1 > 0$. So $L > 0$

From $a_1 b_1 - a_0 \geq 0$, we obtain $R \geq 0$

Substitute condition of b_0 in (13) to ${a_1}^2 b_0 - a_0 a_1 b_1 + {a_0}^2$:

$$ {a_1}^2 b_0 - a_0 a_1 b_1 + {a_0}^2 = {a_1}^2(\sigma^2 + \theta^2) - a_0 a_1 (2\sigma) + {a_0}^2$$

$$= ({a_1}^2 \sigma^2 - 2 a_0 a_1 \sigma + {a_0}^2) + {a_1}^2 \theta^2 = (a_1 \sigma - a_0)^2 + {a_1}^2 \theta^2 > 0 \quad (18)$$

So $C_p, R_p > 0$, and all the components are non-negative.

When $a_1 b_1 \cong a_0$, the value of components are:

$$L = \frac{1}{a_1}, R = 0, C_p = \frac{a_1}{b_0}, R_p = \frac{b_0}{a_0} \quad (19)$$

2. Case non PR case a: $a_0 > 0, a_1 b_1 < a_0$

Similar to non PR case of real pole, we add a positive admittance G_0 to $y_f(s)$

$$y'_f(s) = y_f(s) + G_0 = \frac{a_1 s + a_0}{s^2 + b_1 s + b_0} + G_0 = \frac{G_0 s^2 + (a_1 + b_1 G_0) s + (a_0 + b_0 G_0)}{s^2 + b_1 s + b_0}$$

Now there are two situations: $a_0 > 0$ and $a_0 < 0$, we discuss $a_0 > 0$ in case a and $a_0 < 0$ in case b.

$$y'_f(s) = \frac{G_0 (s^2 + b_0)}{s^2 + b_1 s + b_0} + \frac{(a_1 + b_1 G_0) s + a_0}{s^2 + b_1 s + b_0} = y''_f(s) + y'''_f(s) \quad (20)$$

To decide the value of G_0, we consider PR condition of $y'''_f(s)$. Similar to PR condition of $y_f(s)$, the condition is:

$$(a_1 + b_1 G_0) b_1 \geq a_0 \geq 0 \quad (21)$$

since $a_0 > 0$, so $G_0 \geq \frac{a_0 - a_1 b_1}{b_1^2}$,

If we choose $G_0 = \frac{a_0 - a_1 b_1}{b_1^2}$, so $a_1 + b_1 G_0 = \frac{a_0}{b_1}$ and $G_0 > 0$

Branch 1:

$$z''_f(s) = \frac{1}{y''_f(s)} = \frac{s^2 + b_1 s + b_0}{G_0 (s^2 + b_0)} = \frac{1}{G_0} + \frac{b_1 s}{G_0 (s^2 + b_0)} = \frac{1}{G_0} + \frac{1}{\frac{G_0 s}{b_1} + \frac{G_0 b_0}{b_1 s}} \quad (22)$$

Branch 2:

$$z'''_f(s) = \frac{1}{y'''_f(s)} = \frac{s^2 + b_1 s + b_0}{\frac{a_0}{b_1} s + a_0} = \frac{b_1 s}{a_0} + \frac{b_0}{\frac{a_0}{b_1} s + a_0} = \frac{b_1 s}{a_0} + \frac{1}{\frac{a_0}{b_1 b_0} s + \frac{a_0}{b_0}} \quad (23)$$

The components are:

$$R_1 = \frac{1}{G_0}, C_{1p} = \frac{G_0}{b_1}, L_{1p} = \frac{b_1}{G_0 b_0}, L_2 = \frac{b_1}{a_0}, C_{2p} = \frac{a_0}{b_1 b_0}, R_{2p} = \frac{b_0}{a_0}. \quad (24)$$

It is very easy to prove all of them have non-negative values.

3. Case non PR case b: $a_0 < 0$

$$y'_f(s) = \frac{\left(\frac{a_0}{b_0}+G_0\right)(s^2+b_0)}{s^2+b_1 s+b_0} + \frac{\left(-\frac{a_0}{b_0}\right)s^2+(a_1+b_1G_0)s}{s^2+b_1 s+b_0} = y''_f(s) + y'''_f(s) \quad (25)$$

PR condition of $y''_f(s)$:

$$\frac{a_0}{b_0} + G_0 \geq 0 \text{ or } G_0 \geq \frac{-a_0}{b_0} \quad (26)$$

PR condition of $y'''_f(s)$:

$$(-a_0) \geq 0 \text{ and } G_0 \geq \frac{(-a_0)-a_1 b_1}{b_1^2} \quad (27)$$

Although a_1 can be numerically positive or negative, the appropriate value of G_0 will be determined by the stronger ones of two conditions:

$$G_0 \geq \frac{-a_0}{b_0} \geq 0 \text{ or } G_0 \geq \frac{(-a_0)-a_1 b_1}{b_1^2} \quad (28)$$

This results in two choice of G_0.

Situation 1: $G_0 = \frac{-a_0}{b_0}$.

Then it must satisfy

$$\frac{-a_0}{b_0} \geq \frac{(-a_0)-a_1 b_1}{b_1^2} \text{ or } a_1 b_1 b_0 + a_0 b_0 - a_0 b_1^2 \geq 0 \quad (29)$$

If we assign a new parameter $c_0 = (a_1 b_0 - a_0 b_1)$; then the condition of this situation become:

$$c_0 \geq \frac{(-a_0)b_0}{b_1} \geq 0 \quad (30)$$

Then

$$y'_f(s) = 0 + \frac{\left(-\frac{a_0}{b_0}\right)s^2+\left(a_1-\frac{a_0 b_1}{b_0}\right)s}{s^2+b_1 s+b_0} = \frac{s^2+\frac{c_0}{b_0}s}{s^2+b_1 s+b_0} \quad (31)$$

To decompose the equation:

$$z'_f(s) = \frac{1}{y'_f(s)} = \frac{1}{s} \cdot \frac{s^2+b_1 s+b_0}{\left(-\frac{a_0}{b_0}\right)s+\frac{c_0}{b_0}} = \frac{b_0^2}{c_0} \cdot \frac{1}{s} + \frac{s+\left(b_1+\frac{a_0 b_0}{c_0}\right)}{\left(-\frac{a_0}{b_0}\right)s+\frac{c_0}{b_0}}$$

$$= \frac{b_0^2}{c_0} \cdot \frac{1}{s} + \frac{b_0(b_1 c_0+a_0 b_0)}{c_0^2} + \frac{s\frac{c_0^2+a_0(b_1 c_0+a_0 b_0)}{c_0^2}}{\left(-\frac{a_0}{b_0}\right)s+\frac{c_0}{b_0}} = \frac{1}{Cs} + R + \frac{1}{R_p+1/sL_p} \quad (32)$$

The values of components are:

$$C = \frac{c_0}{b_0^2}, R = \frac{b_0(b_1 c_0+a_0 b_0)}{c_0^2}, L_p = \frac{b_0(c_0^2+a_0 b_1 c_0+a_0^2 b_0)}{c_0^3}, R_p = \frac{b_0(c_0^2+a_0 b_1 c_0+a_0^2 b_0)}{(-a_0)c_0^2} \quad (33)$$

To prove their values are non-negative:

Since $c_0 \geq 0$, $C = \frac{c_0}{b_0^2} \geq 0$.

Since $c_0 \geq \frac{(-a_0)b_0}{b_1}$, then $b_1 c_0 + a_0 b_0 \geq 0$. $R = \frac{b_0(b_1 c_0+a_0 b_0)}{c_0^2} \geq 0$

For L_p and R_p, condition of b_1 and b_0 in (13) are used again:

$$c_0^2 + a_0 b_1 c_0 + a_0^2 b_0 = \left(c_0 + \frac{a_0 b_1}{2}\right)^2 + a_0^2 \left[b_0 - \left(\frac{b_1}{2}\right)^2\right]$$

$$= \left(c_0 + \frac{a_0 b_1}{2}\right)^2 + a_0^2 \left[\sigma^2 + \theta^2 - \left(\frac{2\sigma}{2}\right)^2\right] = \left(c_0 + \frac{a_0 b_1}{2}\right)^2 + a_0^2 \theta^2 > 0 \quad (33)$$

So $L_p, R_p \geq 0$

Situation 2: $G_0 = \frac{(-a_0)-a_1 b_1}{b_1^2}$.

The condition is $\frac{-a_0}{b_0} \leq \frac{(-a_0)-a_1 b_1}{b_1^2}$ or $G_0 + \frac{a_0}{b_0} \geq 0$

Then

$$y'_f(s) = \frac{\left(\frac{a_0}{b_0}+G_0\right)(s^2+b_0)}{s^2+b_1 s+b_0} + \frac{\left(-\frac{a_0}{b_0}\right)s^2+\left(-\frac{a_0}{b_1}\right)s}{s^2+b_1 s+b_0} = y''_f(s) + y'''_f(s);$$

$$\frac{1}{y''_f(s)} = \frac{s^2+b_1 s+b_0}{\left(\frac{a_0}{b_0}+G_0\right)(s^2+b_0)} = \frac{1}{\left(\frac{a_0}{b_0}+G_0\right)} + \frac{b_1 s}{\left(\frac{a_0}{b_0}+G_0\right)(s^2+b_0)} = \frac{b_0}{a_0+b_0 G_0} + \frac{1}{\frac{(a_0+b_0 G_0)s}{b_0 b_1}+\frac{a_0+b_0 G_0}{b_1 s}};$$

$$\frac{1}{y'''_f(s)} = \frac{1}{s}\cdot\frac{s^2+b_1 s+b_0}{\left(-\frac{a_0}{b_0}\right)s+\left(-\frac{a_0}{b_1}\right)} = \frac{b_1 b_0}{(-a_0)}\cdot\frac{1}{s} + \frac{s}{\left(-\frac{a_0}{b_0}\right)s+\left(-\frac{a_0}{b_1}\right)} = \frac{b_1 b_0}{(-a_0)}\cdot\frac{1}{s} + \frac{1}{\left(-\frac{a_0}{b_0}\right)+\left(-\frac{a_0}{b_1}\right)\cdot\frac{1}{s}};$$

$$R_1 = \frac{b_0}{a_0+b_0 G_0}, C_{1p} = \frac{a_0+b_0 G_0}{b_0 b_1}, L_{1p} = \frac{b_1}{a_0+b_0 G_0},$$

$$C_2 = \frac{(-a_0)}{b_1 b_0}, L_{2p} = \frac{b_1}{(-a_0)}, R_{2p} = \frac{b_0}{(-a_0)} \quad (34)$$

APPENDX III All the cases for Brune's Method

We start again from the admittance on 1-port system of poles-residue format

$$y(s) = \frac{r_1}{s-p_1} + \frac{r_2}{s-p_2} + \cdots + \frac{r_{n-1}}{s-p_{n-1}} + \frac{r_n}{s-p_n} + d + hs \qquad (1)$$

Brune process find the sum of all parameters at first:

$$y(s) = \frac{a_n s^n + a_{n-1}s^{n-1} + a_{n-2}s^{n-2} + \cdots + a_0}{b_n s^n + b_{n-1}s^{n-1} + b_{n-2}s^{n-2} + \cdots + b_0} \qquad (2)$$

And then decide

$$y(s) = y_f(s) + y'(s) \qquad (3)$$

where $y_f(s)$ is a component parallel to the port and $y'(s)$ is a less order rational function.

There are totally 8 cases for different patterns of $y_f(s)$ and $y'(s)$.

Case 0: $y(s) = G$

When $y(s)$ equals to a constant, an admittance G is added and it is the end of the process.

Case 1: $b_n = 0$, $y(s)$ has a pole at $s = \infty$

$$y(s) == \frac{a'_{n-1}s^{n-1} + a'_{n-2}s^{n-2} + \cdots + a'_0}{b_{n-1}s^{n-1} + b_{n-2}s^{n-2} + \cdots + b_0} + \frac{a_n}{b_{n-1}}s = y'(s) + As$$

$$a'_i = \begin{cases} a_i - Ab_{i-1} & i = 1, \cdots, n-1 \\ a_0 & i = 0 \end{cases}; A = \frac{a_n}{b_{n-1}}. \quad (4)$$

Case 2: $a_n = 0$, $y(s)$ has a zero at $s = \infty$

$$\frac{1}{y(s)} = \frac{b'_{n-1}s^{n-1} + b'_{n-2}s^{n-2} + \cdots + b'_0}{a_{n-1}s^{n-1} + a_{n-2}s^{n-2} + \cdots + a_0} + \frac{b_n}{a_{n-1}}s = \frac{1}{y'(s)} + As$$

So $y'(s) = \frac{a_{n-1}s^{n-1} + a_{n-2}s^{n-2} + \cdots + a_0}{b'_{n-1}s^{n-1} + b'_{n-2}s^{n-2} + \cdots + b'_0}; A = \frac{b_n}{a_{n-1}}, b'_i = \begin{cases} b_i - Aa_{i-1} & i = 1, \cdots, n-1 \\ b_0 & i = 0 \end{cases}$. (5)

Case 3: $b_0 = 0$, $y(s)$ has a pole at $s = 0$

$$y(s)=\frac{a_n's^{n-1}+a_{n-1}'s^{n-2}+a_{n-2}'s^{n-3}+\cdots+a_1'}{b_ns^{n-1}+b_{n-1}s^{n-2}+b_{n-2}s^{n-3}+\cdots+b_1}+\frac{a_0}{b_1s}=y'(s)+\frac{A}{s}$$

$$a_i'=\begin{cases}a_i-Ab_{i+1} & i=1,\cdots,n-1\\ a_n & i=n\end{cases};A=\frac{a_0}{b_1}. \qquad (6)$$

Case 4: $a_0=0$, $y(s)$ has a zero at $s=0$

$$\frac{1}{y(s)}=\frac{b_n's^{n-1}+b_{n-1}'s^{n-2}+b_{n-2}'s^{n-3}+\cdots+b_1'}{a_ns^{n-1}+a_{n-1}s^{n-2}+a_{n-2}s^{n-3}+\cdots+a_1}+\frac{b_0}{a_1s}=\frac{1}{y'(s)}+\frac{A}{s},$$

$$A=\frac{b_0}{a_1};b_i'=\begin{cases}b_i-Aa_{i+1} & i=1,\cdots,n-1\\ b_n & i=n\end{cases}. \qquad (7)$$

Case 5: $Q(s)|_{s=\pm j\omega_p}=0$, $y(s)$ has poles on imaginary axis

In this case, $\left(s^2+\omega_p^2\right)$ are separated from $Q(s)$ at first:

$$Q(s)=\left(s^2+w_p^2\right)(b_{n-2}'s^{n-2}+b_{n-3}'s^{n-3}+\cdots+b_1's+b_0') \qquad (8)$$

The next step is to find residue of k for $s=\pm j\omega_p$:

$$k=\left[\frac{P(s)}{Q'(s)}\right]_{s=\pm j\omega_p}=\left[\frac{M_1}{N_2'}\right]_{s=\pm j\omega_p}=\left[\frac{N_1}{M_2'}\right]_{s=\pm j\omega_p}$$

$$y(s)=\frac{a_{n-2}''s^{n-2}+a_{n-3}''s^{n-3}+\cdots+a_1''s+a_0''}{b_{n-2}'s^{n-2}+b_{n-3}'s^{n-3}+\cdots+b_1's+b_0'}+\frac{2ks}{s^2+\omega_p^2}=y'(s)+\frac{1}{\frac{s}{2k}+\frac{w_p^2}{2k}\cdot\frac{1}{s}}. \qquad (9)$$

Case 6: $P(s)|_{s=\pm j\omega_p}=0$, $y(s)$ has zeros on imaginary axis

Similar to case 5,

$$P(s)=\left(s^2+\omega_p^2\right)(a_{n-2}'s^{n-2}+a_{n-3}'s^{n-3}+\cdots+a_1's+a_0') \qquad (10)$$

And residue can be gotten through

$$k=\left[\frac{Q(s)}{P'(s)}\right]_{s=\pm j\omega_p}=\left[\frac{M_2}{N_1'}\right]_{s=\pm j\omega_p}=\left[\frac{N_2}{M_1'}\right]_{s=\pm j\omega_p}$$

$$\frac{1}{y(s)} = \frac{b''_{n-2}s^{n-2}+b''_{n-3}s^{n-3}+\cdots+b''_1 s+b''_0}{a'_{n-2}s^{n-2}+a'_{n-3}s^{n-3}+\cdots+a'_1 s+a'_0} + \frac{2ks}{s^2+\omega_p^2} = \frac{1}{y'(s)} + \frac{1}{\frac{s}{2k}+\frac{w_p^2}{2k}\cdot\frac{1}{s}} \quad (11)$$

Case 7: When $y(s)$ doesn't fit to other cases, it will be in case 7.

This case involves two steps; the first step is to find:

$$R = Re[y(s)]_{min}|_{s=j\omega_p} \ , \omega_p \epsilon [0,\infty) \quad (12)$$

The second step is to substrate $y(j\omega_p)$ from $y(s)$:

$$y_p(s) = y(s) - y(j\omega_p) \quad (13)$$

Depends on different ω_p, $y_p(s)$ can be applied methods from case 2,4,6.

1) $\omega_p = \infty \rightarrow y(j\omega_p) = \frac{a_n}{b_n}$

$$y(s) - y(j\omega_p) = \frac{a_n s^n+a_{n-1}s^{n-1}+a_{n-2}s^{n-2}+\cdots+a_0}{b_n s^n+b_{n-1}s^{n-1}+b_{n-2}s^{n-2}+\cdots+b_0} - \frac{a_n}{b_n} = \frac{a'_{n-1}s^{n-1}+a'_{n-2}s^{n-2}+\cdots+a'_0}{b_n s^n+b_{n-1}s^{n-1}+\cdots+b_0}$$

$$a'_i = \begin{cases} a_i - \frac{a_n}{b_n}b_i & i = 0,\cdots,n-1 \\ 0 & i = n \end{cases} \quad (14)$$

Then the format is same to case 2.

2) $\omega_p = 0 \rightarrow y(j\omega_p) = \frac{a_0}{b_0}$

$$y(s) - y(j\omega_p) = \frac{a_n s^n+a_{n-1}s^{n-1}+a_{n-2}s^{n-2}+\cdots+a_0}{b_n s^n+b_{n-1}s^{n-1}+b_{n-2}s^{n-2}+\cdots+b_0} - \frac{a_0}{b_0} = \frac{a'_n s^n+a'_{n-1}s^{n-1}+\cdots+a'_1 s}{b_n s^n+b_{n-1}s^{n-1}+\cdots+b_0}$$

$$a'_i = \begin{cases} a_i - \frac{a_0}{b_0}b_i & i = 1,\cdots,n \\ 0 & i = 0 \end{cases} \quad (15)$$

Then the format is same to case 4.

3) $\omega_p = \omega_p \rightarrow y(j\omega_p) = Re[y(j\omega_p)]_{min} + j\, Im[y(j\omega_p)]$

Type 1: $Im[y(j\omega_p)] < 0$

$$y(s)_{s=j\omega_p} = Re[y(j\omega_p)] + s\ \frac{Im[y(j\omega_p)]}{\omega_p} = A + Bs, A > 0, B < 0 \quad (16)$$

$$y(s) - y(j\omega_p) = \frac{a_n s^n + a_{n-1}s^{n-1} + a_{n-2}s^{n-2} + \cdots + a_0}{b_n s^n + b_{n-1}s^{n-1} + b_{n-2}s^{n-2} + \cdots + b_0} - A + (-B)s$$

$$= \frac{a'_{n+1}s^{n+1}+a'_n s^n+\cdots+a'_1 s+a'_0}{b_n s^n+b_{n-1}s^{n-1}+\cdots+b_0} = \frac{P_1(s)}{Q(s)}. \quad (17)$$

Now $P_1(s)$ must have zero pairs at $s = \pm j\omega_p$, it matches case 6 that:

$$P_1(s) = \left(s^2 + \omega_p^2\right)\left(a''_{n-1}s^{n-1} + a''_{n-2}s^{n-2} + \cdots + a''_1 s + a''_0\right) \quad (18)$$

$$\frac{1}{y(s)-y(jw_p)} = \frac{Q(s)}{P_1(s)} = \frac{b''_{n-2}s^{n-2}+b''_{n-3}s^{n-3}+\cdots+b''_1 s+b''_0}{a''_{n-1}s^{n-1}+a''_{n-2}s^{n-2}+\cdots+a''_1 s+a''_0} + \frac{2ks}{s^2+w_p^2} \quad (19)$$

$$y_1(s) = \frac{a''_{n-1}s^{n-1}+a''_{n-2}s^{n-2}+\cdots+a''_1 s+a''_0}{b''_{n-2}s^{n-2}+b''_{n-3}s^{n-3}+\cdots+b''_1 s+b''_0} = \frac{a'''_{n-2}s^{n-2}+a'''_{n-2}s^{n-3}+\cdots+a'''_1 s+a'''_0}{b''_{n-2}s^{n-2}+b''_{n-3}s^{n-3}+\cdots+b''_1 s+b''_0} + Ds = y'(s) + Ds$$

(20)

So the total transformation for $y(s)$ in this case is:

$$\frac{1}{y(s)-A-Bs} = \frac{1}{y'(s)+Ds} + \frac{1}{\frac{s}{2k}+\frac{w_p^2}{2k}\cdot\frac{1}{s}} \quad (21)$$

And components are:

$$R = \frac{1}{A}, C_1 = B = \frac{Im[y(jw_p)]}{w_p}, C_2 = \frac{1}{2k}, L = \frac{2k}{w_p^2}, C_3 = D = \frac{-C_1C_2}{C_1+C_2},$$

$$C = C_1 + C_2, n = \frac{C_2}{C_1+C_2} = \sqrt{\frac{C_2+C_3}{C_1+C_2}}. \quad (22)$$

Type 2: $Im\left[y\left(jw_p\right)\right] > 0$

$$y(s)_{s=jw_p} = Re\left[y\left(jw_p\right)\right]_{min} + \frac{-Im[y(jw_p)]\cdot w_p}{s} = A + \frac{B}{s}, A > 0, B < 0 \quad (23)$$

Instead of $y(s) - y\left(jw_p\right)$, we do

$$\frac{1}{y(s)-A} - \frac{1}{\frac{B}{s}} = \frac{b'_{n+1}s^{n+1}+b'_n s^n+\cdots+b'_1 s+b'_0}{a'_n s^n+a'_{n-1}s^{n-1}+\cdots+a'_0} = \frac{\left(s^2+w_p^2\right)\left(b''_{n-1}s^{n-1}+b''_{n-2}s^{n-2}+\cdots+b''_1 s+b''_0\right)}{a'_n s^n+a'_{n-1}s^{n-1}+\cdots+a'_0} \quad (24)$$

$$\frac{1}{\frac{1}{y(s)-A}-\frac{1}{\frac{B}{s}}} = \frac{2ks}{s^2+w_p^2} + \frac{a''_{n-2}s^{n-2}+a''_{n-3}s^{n-3}+\cdots+a''_1 s+a''_0}{b''_{n-1}s^{n-1}+b''_{n-2}s^{n-2}+\cdots+b''_1 s+b''_0} = \frac{2ks}{s^2+w_p^2} + \frac{1}{\frac{1}{y'(s)}+Ds} \quad (25)$$

Then components are

$$R = \frac{1}{A}, L_1 = \frac{1}{B} = \frac{1}{-Im\left[y\left(jw_p\right)\right]\cdot w_p}, L_2 = \frac{1}{2k}, C = \frac{2k}{w_p^2},$$

$$L_3 = D = \frac{-L_1L_2}{L_1+L_2}, n = \frac{L_2}{\sqrt{(L_1+L_2)(L_2+L_3)}}, L'_1 = L_1+L_2, L'_2 = L_2+L_3 \quad (26)$$

APPENDX IV PR property check for pole-residue components

This appendix will check the PR property for sum of two pole-residue components $y_{fn}(s)+y_{fp}(s)$, where $y_{fn}(s)$ is non PR and $y_{fp}(s)$ is PR. Since each of them has two possible cases: real case and complex conjugate case. Their sum is divided into three patterns:

A. $y_{fn}(s)$ and $y_{fp}(s)$ are all real case, $y_{fn}(s)+y_{fp}(s)=y_{fnr}(s)+y_{fpr}(s)$

$$y_{fnr}(s)=\frac{a}{s+b}, a<0, b\geq 0; y_{fpr}(s)=\frac{c}{s+d}, c>0, d\geq 0 \quad (1)$$

$$y_{fnr}(s)+y_{fpr}(s)=\frac{a}{s+b}+\frac{c}{s+d}=\frac{(a+c)s+(ad+bc)}{s^2+(b+d)s+bd}=\frac{\Lambda_1 s+\Lambda_0}{s^2+\Gamma_1 s+\Gamma_0} \quad (2)$$

Based on [70], RR conditions for $y_{fnr}(s)+y_{fpr}(s)$ are:

$$\Lambda_1,\Lambda_0,\Gamma_1,\Gamma_0\geq 0 \text{ and } \Lambda_1\Gamma_1\geq\Gamma_0 \quad (3)$$

Since

$$\Gamma_1=b+d\geq 0 \text{ and } \Gamma_0=bd\geq 0 \quad (4)$$

Then

$$\Lambda_1\geq\Gamma_0/\Gamma_1\geq 0 \quad (5)$$

To satisfy the rest conditions:

$$\Gamma_0\geq 0 \text{ and } \Lambda_1\Gamma_1\geq\Gamma_0 \quad (6)$$

Substitute a,b,c,d to (6), they become:

$$ad+bc\geq 0 \text{ and } (a+c)(b+d)\geq ad+bc \quad (7)$$

Simplify the first equation:

$$ad+bc\geq 0 \ll=\gg bc\geq -ad \ll=\gg \frac{c}{|a|}\geq\frac{d}{b} \quad (8)$$

Simplify the second equation:

$$ab+ad+bc+cd\geq ad+bc \ll=\gg ab+cd\geq 0 \ll=\gg cd\geq -ab \quad (9)$$

$$\frac{c}{|a|}\geq\frac{b}{d} \quad (10)$$

The final conditions for case A are:

1) $b,d\neq 0$
2) $\frac{c}{|a|}\geq\frac{b}{d}$
3) $\frac{c}{|a|}\geq\frac{d}{b}$

B. One of $y_{fn}(s)$ and $y_{fp}(s)$ is real, the other is complex conjugate. Then

$$y_{fn}(s)=y_{fnr}(s)=\frac{c}{s+d} \text{ and } y_{fp}(s)=y_{fpc}(s)=\frac{a_1 s+a_0}{s^2+b_1 s+b_0};$$

$$\text{or } y_{fn}(s)=y_{fnc}(s)=\frac{a_1 s+a_0}{s^2+b_1 s+b_0} \text{ and} y_{fp}(s)=y_{fpr}(s)=\frac{c}{s+d} \quad (11)$$

These two are similar to each other, and their sum:

$$y_{fn}(s)+y_{fp}(s)=\frac{(a_0+c_1)s^2+(a_0d_1+b_0c_1+c_0)s+(a_0d_0+b_0c_0)}{s^3+(b_0+d_1)s^2+(b_0d_1+d_0)s+b_0d_0}=\frac{\Lambda_2s^2+\Lambda_1s+\Lambda_0}{s^3+\Gamma_2s^2+\Gamma_1s+\Gamma_0}=\frac{p(s)}{q(s)} \quad (12)$$

Based on [70], if $y(s)=\frac{p(s)}{q(s)}$ is PR it must satisfy two conditions:

a) $p(s)+q(s)$ has no roots in ORHP

$$p(s)+q(s)=s^3+(a_0+b_0+c_1+d_1)s^2+(a_0d_1+b_0c_1+b_0d_1+c_0+d_0)s+(a_0d_0+b_0c_0+b_0d_0)=s^3+\psi_2s^2+\psi_1s+\psi_0 \quad (13)$$

The condition that $p(s)+q(s)$ has no roots in ORHP are:

$$\psi_i\geq 0,\psi_1\psi_2\geq\psi_0 \quad (14)$$

b) $Re[y(j\omega)]\geq 0$

Based on Chapter 1 of [56], $y(s)$ can be transformed into:

$$y(s)=\frac{p(s)}{q(s)}=\frac{m_1(s)+n_1(s)}{m_2(s)+n_2(s)}=\frac{(\Lambda_2s^2+\Lambda_0)+(\Lambda_1s)}{(\Gamma_2s^2+\Gamma_0)+(s^3+\Gamma_1s)} \quad (15)$$

Then

$$Re[y(j\omega)]=\frac{m_1(s)m_2(s)-n_1(s)n_2(s)}{m_2(s)^2-n_2(s)^2}\Big|_{s=j\omega} \quad (16)$$

To prove denominator of $Re[y(j\omega)]$ is positive:

$$m_2(s)^2-n_2(s)^2|_{s=j\omega}=(\Gamma_2s^2+\Gamma_0)^2-s^2(s^2+\Gamma_1)^2$$
$$=(-\Gamma_2\omega^2+\Gamma_0)^2+\omega^2(-\omega^2+\Gamma_1)^2\geq 0 \quad (17)$$

Seek conditions for divider, we suppose a parameter $\Omega=\omega^2>0$:

$m_1(s)m_2(s)-n_1(s)n_2(s)|_{s=j\omega}$

$=(\Lambda_2s^2+\Lambda_0)(\Gamma_2s^2+\Gamma_0)-\Lambda_1s(s^3+\Gamma_1s)$

$=(\Lambda_2\Gamma_2s^4+\Lambda_2\Gamma_0s^2+\Lambda_0\Gamma_2s^2+\Lambda_0\Gamma_0)-(\Lambda_1s^4+\Lambda_1\Gamma_1s^2)$

$=(\Lambda_2\Gamma_2-\Lambda_1)s^4+(\Lambda_2\Gamma_0+\Lambda_0\Gamma_2-\Lambda_1\Gamma_1)s^2+\Lambda_0\Gamma_0$

$=(\Lambda_2\Gamma_2-\Lambda_1)\omega^4+(\Lambda_1\Gamma_1-\Lambda_2\Gamma_0-\Lambda_0\Gamma_2)\omega^2+\Lambda_0\Gamma_0$

$=(a_0b_0+c_1d_1-c_0)\omega^4+[a_0b_0(d_1^2-2d_0)+b_0^2(c_1d_1-c_0)+c_0d_0]\omega^2+(a_0b_0d_0^2+b_0^2c_0d_0)$

$=\Phi_2\Omega^2+\Phi_1\Omega+\Phi_0$ (18)

The condition for $\Phi_2\Omega^2+\Phi_1\Omega+\Phi_0\geq 0$ and $\Omega\geq 0$ are:

$$\Phi_2\geq 0,\Phi_0\geq 0 \text{ and } 2\sqrt{\Phi_0\Phi_2}\geq-\Phi_1 \quad (19)$$

Conclude the final conditions for case B:

1) Calculate:

$\psi_2=a_1+b_1+c+d;\ \psi_1=b_1c+a_1d+b_1d+a_0+b_0;\psi_0=b_0c+a_0d+b_0d;$

Then ψ must satisfy $\psi_0 \geq 0, \psi_1 \geq 0, \psi_2 \geq 0$ and $\psi_1\psi_2 \geq \psi_0$

2) Calculate:

$\Phi_2 = cd + (a_1b_1 - a_0); \Phi_1 = cd(b_1^2 - 2b_0) + d^2(a_1b_1 - a_0) + a_0b_0;$

$\Phi_0 = b_0^2cd + a_0b_0d^2$

Then, then Φ must satisfy $\Phi_2 \geq 0$, $\Phi_0 \geq 0$ and $2\sqrt{\Phi_0\Phi_2} \geq -\Phi_1$

When these conditions are satisfied, then

$$y_{fn}(s) + y_{fr}(s) = \frac{(a_0+c_1)s^2+(a_0d_1+b_0c_1+c_0)s+(a_0d_0+b_0c_0)}{s^3+(b_0+d_1)s^2+(b_0d_1+d_0)s+b_0d_0} \quad (20)$$

is PR.

C. The last case is that $y_{fn}(s)$ and $y_{fp}(s)$ are all complex conjugate. Then suppose

$$y_{fn}(s) = y_{fnc}(s) = \frac{a_1s+a_0}{s^2+b_1s+b_0} \text{ and } y_{fp}(s) = y_{fpc}(s) = \frac{c_1s+c_0}{s^2+d_1s+d_0} \quad (21)$$

Their sum,

$$y_{fn}(s) + y_{fp}(s) = \frac{(a_1+c_1)s^3+(a_0+c_0+a_1d_1+b_1c_1)s^2+(a_1d_0+a_0d_1+b_0c_1+b_1c_0)s+(a_0d_0+b_0c_0)}{s^4+(b_1+d_1)s^3+(b_0+d_0+b_1d_1)s^2+(b_0d_1+b_1d_0)s+b_0d_0}$$

$$= \frac{\Lambda_3s^3+\Lambda_2s^2+\Lambda_1s+\Lambda_0}{s^4+\Gamma_3s^3+\Gamma_2s^2+\Gamma_1s+\Gamma_0} = \frac{p(s)}{q(s)}; \quad (22)$$

$\Lambda_0 = a_0d_0 + b_0c_0; \Lambda_1 = a_1d_0 + a_0d_1 + b_0c_1 + b_1c_0;$

$\Lambda_2 = a_0 + c_0 + a_1d_1 + b_1c_1; \Lambda_3 = a_1 + c_1;$

$\Gamma_0 = b_0d_0; \Gamma_1 = b_0d_1 + b_1d_0; \Gamma_2 = b_0 + d_0 + b_1d_1; \Gamma_3 = b_1 + d_1$

Based on [70], if $y(s) = \frac{p(s)}{q(s)}$ is PR it must satisfy two conditions:

a. $p(s) + q(s)$ has no roots in ORHP

$$p(s) + q(s) = s^4 + (\Lambda_3 + \Gamma_3)s^3 + (\Lambda_2 + \Gamma_2)s^2 + (\Lambda_1 + \Gamma_1)s + (\Lambda_0 + \Gamma_0)$$

$$= s^4 + \psi_3s^3 + \psi_2s^2 + \psi_1s + \psi_0 \quad (23)$$

The condition that $p(s) + q(s)$ has no roots in ORHP are:

$$\psi_i \geq 0, \psi_1\psi_2\psi_3 \geq \psi_0\psi_3^2 + \psi_1^2 \text{ and } \psi_2^2 \geq 4\psi_0 \quad (24)$$

b. $Re[y(j\omega)] \geq 0$

Transform $y(s)$ based on [56]:

$$y(s) = \frac{\Lambda_3s^3+\Lambda_2s^2+\Lambda_1s+\Lambda_0}{s^4+\Gamma_3s^3+\Gamma_2s^2+\Gamma_1s+\Gamma_0} = \frac{p(s)}{q(s)} = \frac{m_1(s)+n_1(s)}{m_2(s)+n_2(s)} = \frac{(\Lambda_2s^2+\Lambda_0)+(\Lambda_3s^3+\Lambda_1s)}{(s^4+\Gamma_2s^2+\Gamma_0)+(\Omega_3s^3+\Gamma_1s)},$$

$$Re[y(j\omega)] = \frac{m_1(s)m_2(s)-n_1(s)n_2(s)}{m_2(s)^2-n_2(s)^2}\Big|_{s=j\omega} \quad (25)$$

To prove the denominator is positive

$$m_2(s)^2 - n_2(s)^2|_{s=j\omega} = (s^4 + \Gamma_2 s^2 + \Gamma_0)^2 - s^2(\Gamma_3 s^2 + \Gamma_1)^2$$
$$= (\omega^4 - \Gamma_2\omega^2 + \Gamma_0)^2 + \omega^2(-\Gamma_3\omega^2 + \Gamma_1)^2 \geq 0. \quad (26)$$

Seek conditions for the $m_1(s)m_2(s) - n_1(s)n_2(s) \geq 0$, we suppose a parameter $\Omega = \omega^2 > 0$:

$m_1(s)m_2(s) - n_1(s)n_2(s)|_{s=j\omega}$

$= (\Lambda_2 s^2 + \Lambda_0)(s^4 + \Gamma_2 s^2 + \Gamma_0) - (\Lambda_3 s^3 + \Lambda_1 s)(\Gamma_3 s^3 + \Gamma_1 s)$

$= (\Lambda_2 s^6 + \Lambda_2\Gamma_2 s^4 + \Lambda_2\Gamma_0 s^2 + \Lambda_0 s^4 + \Lambda_0\Gamma_2 s^2 + \Lambda_0\Gamma_0) - (\Lambda_3\Gamma_3 s^6 + \Lambda_3\Gamma_1 s^4 + \Lambda_1\Gamma_3 s^4 + \Lambda_1\Gamma_1 s^2)$

$= (\Lambda_2 - \Lambda_3\Gamma_3)s^6 + (\Lambda_2\Gamma_2 + \Lambda_0 - \Lambda_3\Gamma_1 - \Lambda_1\Gamma_3)s^4 + (\Lambda_2\Gamma_0 + \Lambda_0\Gamma_2 - \Lambda_1\Gamma_1)s^2 + \Lambda_0\Gamma_0$

$= (\Lambda_3\Gamma_3 - \Lambda_2)\omega^6 + (\Lambda_2\Gamma_2 + \Lambda_0 - \Lambda_3\Gamma_1 - \Lambda_1\Gamma_3)\omega^4 + (\Lambda_1\Gamma_1 - \Lambda_2\Gamma_0 - \Lambda_0\Gamma_2)\omega^2 + \Lambda_0\Gamma_0$

$= \Phi_3\Omega^3 + \Phi_2\Omega^2 + \Phi_1\Omega + \Phi_0$. (27)

Based on [70], the condition for $\Phi_3\Omega^3 + \Phi_2\Omega^2 + \Phi_1\Omega + \Phi_0 \geq 0$ are:

I) $\Phi_3 = 0, \Phi_2 \geq 0, \Phi_0 \geq 0$ *and* $\Phi_1 \geq -2\sqrt{\Phi_0\Phi_2}$ (28)

II) $\Phi_3 \geq 0, \Phi_0 \geq 0$ and two conditions holds:

$$\Phi_1 \geq 0 \text{ and } \Phi_2 \geq -\sqrt{3\Phi_1\Phi_3}$$

Or $\Phi_2^2 > 3\Phi_1\Phi_3$ and $2\Phi_2^3 - 9\Phi_1\Phi_2\Phi_3 + 27\Phi_0\Phi_2^2 \geq 2(\Phi_2^2 - 3\Phi_1\Phi_3)^{3/2}$ (29)

Conclude the final conditions for case C:

1) Calculate:

$\psi_0 = a_0d_0 + b_0c_0 + b_0d_0$;

$\psi_1 = a_1d_0 + a_0d_1 + b_0c_1 + b_1c_0 + b_0d_1 + b_1d_0$;

$\psi_2 = a_0 + b_0 + c_0 + d_0 + a_1d_1 + b_1c_1 + b_1d_1$;

$\psi_3 = a_1 + b_1 + c_1 + d_1$;

Then ψ must satisfy:

$\psi_0 \geq 0, \psi_1 \geq 0, \psi_2 \geq 0, \psi_3 \geq 0, \psi_1\psi_2\psi_3 \geq \psi_0\psi_3^2 + \psi_1^2$ and $\psi_2^2 \geq 4\psi_0$.

2) Calculate:

$\Phi_0 = a_0b_0d_0^2 + b_0^2c_0d_0$;

$\Phi_1 = a_0b_0d_1^2 + b_0^2c_1d_1 + a_1b_1d_0^2 + b_1^2c_0d_0 - 2a_0b_0d_0 - 2b_0c_0d_0 - b_0^2c_0 - a_0d_0^2$;

$\Phi_2 = (a_0b_0 + c_0d_0) + 2(a_0d_0 + b_0c_0) + a_1b_1(d_1^2 - 2d_0) + c_1d_1(b_1^2 - 2b_0) - b_1^2c_0 - a_0d_1^2$;

$\Phi_3 = a_1b_1 + c_1d_1 - a_0 - c_0$;

Then, then Φ must satisfy i) and ii)

a) $\Phi_3 = 0, \Phi_2 \geq 0, \Phi_0 \geq 0$ *and* $\Phi_1 \geq -2\sqrt{\Phi_0\Phi_2}$;

b) $\Phi_3 \geq 0, \Phi_0 \geq 0$ and two conditions holds:

$\Phi_1 \geq 0$ *and* $\Phi_2 \geq -\sqrt{3\Phi_1\Phi_3}$

Or $\Phi_2^2 > 3\Phi_1\Phi_3$ and $2\Phi_2^3 - 9\Phi_1\Phi_2\Phi_3 + 27\Phi_0\Phi_2^2 \geq 2(\Phi_2^2 - 3\Phi_1\Phi_3)^{3/2}$.

When these conditions are satisfied, then

$$y_{fn}(s) + y_{fp}(s) = \frac{(a_1+c_1)s^3+(a_0+c_0+a_1d_1+b_1c_1)s^2+(a_1d_0+a_0d_1+b_0c_1+b_1c_0)s+(a_0d_0+b_0c_0)}{s^4+(b_1+d_1)s^3+(b_0+d_0+b_1d_1)s^2+(b_0d_1+b_1d_0)s+b_0d_0} \quad (30)$$

is PR.

REFERENCE

[1] Myers, C.J.; Barker, N.; Kuwahara, H.; Jones, K.; Madsen, C.; Nguyen, N.-P.D.; , "Genetic design automation," *Computer-Aided Design - Digest of Technical Papers, 2009. ICCAD 2009. IEEE/ACM International Conference on* , vol., no., pp.713-716, 2-5 Nov. 2009

[2] C. R. Paul, *Analysis of Multiconductor Transmission Lines.* New York: Wiley, 1994.

[3] R. Achar and M. Nakhla, "Simulation of high-speed interconnects," *Proceedings IEEE,* vol. 89, no. 5, pp. 693-728, May 2001.

[4]Waldner, Jean-Baptiste (2008). *Nanocomputers and Swarm Intelligence*. London: ISTE John Wiley & Sons. p. 205. ISBN 1-84821-009-4.

[5] R. Ghodssi, P. Lin (2011). *MEMS Materials and Processes Handbook*. Berlin:Springer. ISBN 978-0-387-47316-1.

[6] Steinecke, T.; Koehne, H.; Schmidt, M.; , "Behavioral EMI models of complex digital VLSI circuits," *Electromagnetic Compatibility, 2003. EMC '03.*

[7]Ying Liu; Pileggi, L.T.; Strojwas, A.J., "Model order-reduction of RC(L) interconnect including variational analysis," *Design Automation Conference, 1999. Proceedings. 36th* , vol., no., pp.201,206, 1999

[8]Odabasioglu, A.; Celik, M.; Pileggi, L.T., "PRIMA: passive reduced-order interconnect macromodeling algorithm," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on* , vol.17, no.8, pp.645,654, Aug 1998

[9] Nagel, L. W, and Pederson, D. O., *SPICE (Simulation Program with Integrated Circuit Emphasis)*, Memorandum No. ERL-M382, University of California, Berkeley, Apr. 1973

[10] Gordon W. Roberts, Adel S. Sedra, *Spice,*2nd ed. Oxford University Press, Oxford, 1997

[11] David M.Pozar, *Microwave Engineering*, 3rd ed. New York: Wiley, 2009

[12] Dyer, S.A.; Xin He; , "Least-squares fitting of data by polynomials," *Instrumentation & Measurement Magazine, IEEE* , vol.4, no.4, pp.46-51, Dec 2001

[13] Aimin Yang; Wang Ai-ling; Jincai Chang; , "The research on parallel least squares curve fitting algorithm," *Test and Measurement, 2009. ICTM '09. International Conference on* , vol.2, no., pp.201-204, 5-6 Dec. 2009

[14] Pillage, L.T.; Rohrer, R.A.; "Asymptotic waveform evaluation for timing analysis," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on* , vol.9, no.4, pp.352-366, Apr 1990

[15]Eli Chiprout, Michel S. Nakhla, *Asymptotic Waveform Evaluation*. Springer Science+Business Media New York, 1994

[16] E. Chiprout and M. Nakhla, *Asymptotic Waveform Evaluation and Moment Matching*

for Interconnect Analysis. Boston: Kluwer Academic Publishers, 1993.

[17] P. Feldmann and R. W. Freund, “Efficient linear circuit analysis by Padé approximation via the Lanczos process,” *IEEE Trans. Computer-Aided Design*, vol. 14, pp. 639–649, May 1995.

[18] Dharmendra Saraswat,*Passive Macromodeling of Linear Subnetworks Characterized by Measured Simulated Data,*Master thesis, Carleton University, Ottawa, 2003

[19] Dharmendra Saraswat, *Global Compact Passive Macromodeling Algorithms for High-Speed Circuits,* Phd thesis**,** Carleton University, Ottawa, 2006

[20] B. Gustavsen and A. Semlyen, “Rational approximation of frequency domain responses by vector fitting,” *IEEE Trans. Power Del.*, vol. 14, no. 3, pp. 1052–1061, Jul. 1999.

[21] A. Semlyen and B. Gustavsen, “Vector fitting by pole relocation for the state equation approximation of nonrational transfer matrices,” *Circuits Syst. Signal Process.*, vol. 19, no. 6, pp. 549–566, Nov. 2000.

[22] C. K. Sanathanan and J. Koerner, “Transfer function synthesis as a ratioof two complex polynomials,” *IEEE Trans. Autom. Control*, vol. AC-8, no. 1, pp. 56–58, Jan. 1963.

[23] B. Gustavsen and A. Semlyen, “Simulation of transmission line transients using vector fitting and modal decomposition,” *IEEE Trans. Power Del.*, vol. 13, no. 2, pp. 605–614, Apr. 1998.

[24] B. Gustavsen, “Relaxed vector fitting algorithm for rational approximation of frequency domain responses,” in *Proc. 10th IEEE Workshop Signal Propagation Interconnects*, Berlin, Germany, May 9–12, 2006, pp. 97–100.

[25] S. Grivet-Talocia and M. Bandinu, “Improving the convergence of vector fitting for equivalent circuit extraction from noisy frequency responses,” *IEEE Trans. Electromagn. Compat.*, vol. 48, no. 1, pp. 104–120, Feb. 2006.

[26] R. Gao, Y. S. Mekonnen, W. T. Beyene, and J. E. Schutt-Ainé, “Black-box modeling of passive systems by rational function approximation,” *IEEE Trans. Adv. Packag.*, vol. 28, no. 2, pp. 209–215, May 2005.

[27] B. Gustavsen, “Improving the pole relocating properties of vector fitting,” *IEEE Trans. Power Delivery*, vol. 21, no. 3, pp. 1587–1592, Aug. 2006.

[28] Deschrijver, D.; Mrozowski, M.; Dhaene, T.; De Zutter, D., "Macromodeling of Multiport Systems Using a Fast Implementation of the Vector Fitting Method," *Microwave and Wireless Components Letters, IEEE* , vol.18, no.6, pp.383,385, June 2008

[29] Triverio, P.; Grivet-Talocia, S.; Nakhla, M.S., "An improved fitting algorithm for parametric macromodeling from tabulated data," *Signal Propagation on Interconnects, 2008. SPI 2008. 12th IEEE Workshop on* , vol., no., pp.1,4, 12-15 May 2008

[30] Chinea, A.; Grivet-Talocia, S., "A parallel Vector Fitting implementation for fast macromodeling of highly complex interconnects," *Electrical Performance of Electronic Packaging and Systems (EPEPS), 2010 IEEE 19th Conference on* , vol., no., pp.101,104, 25-27 Oct. 2010

[31] Chinea, A.; Grivet-Talocia, S., "On the Parallelization of Vector Fitting Algorithms," *Components, Packaging and Manufacturing Technology, IEEE Transactions on* , vol.1, no.11, pp.1761,1773, Nov. 2011

[32] Coelho, C.P.; Phillips, J.R.; Silveira, L.M.; , "Robust rational function approximation algorithm for model generation," *Design Automation Conference, 1999. Proceedings. 36th* , vol., no., pp.207-212, 1999

[33] Elzinga, M.; Virga, K.L.; Zhao, L.; Prince, J.L.; , "Pole-residue formulation for transient simulation of high-frequency interconnects using householder LS curve-fitting techniques," *Advanced Packaging, IEEE Transactions on* , vol.23, no.2, pp.142-147, May 2000

[34] Schutt-Aine, J.E.; Mittra, R.; , "Scattering parameter transient analysis of transmission lines loaded with nonlinear terminations," *Microwave Theory and Techniques, IEEE Transactions on* , vol.36, no.3, pp.529-536, Mar 1988

[35] Komuro, T.; "Time-domain analysis of lossy transmission lines with arbitrary terminal networks," *Circuits and Systems, IEEE Transactions on* , vol.38, no.10, pp.1160-1164, Oct 1991

[36] F. Ferranti, L. Knockaert, T. Dhaene, "Parameterized S-Parameter Based Macromodeling With Guaranteed Passivity," *IEEE Microwave and Wireless Components Letters*, vol. 19, no. 10, pp. 608-610, Oct. 2009.

[37] J. Phillips, L. Daniel, and L. M. Silveira, "Guaranteed passive balancing transformations for model order reduction," *IEEE Trans. Computer-Aided Design*, vol. 22, pp. 1027–1041, Aug. 2003.

[38] Coelho, C.P.; Phillips, J.; Silveira, L.M., "A convex programming approach for generating guaranteed passive approximations to tabulated frequency-data," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on* , vol.23, no.2, pp.293,301, Feb. 2004

[39] L. Vandenberghe, "Interior-point methodsfor semidefinite programming problems in signal processing and control," presented at the SIAM Conf. Linear Algebra Signals, Syst., Contr., 2001.

[40] L. Daniel and J. Phillips, "Model order reduction for strictly passive and causal distributed systems," in *Proc. 39th Design Automation Conf.*, New Orleans, LA, June 2002, pp. 46–51.

[41] G. H. Golub and C. F. V. Loan, *Matrix Computations*. Baltimore, MD: Johns Hopkins Univ. Press, 1989.

[42] Min Ma; Khazaka, R.; , "Model Order Reduction With Parametric Port Formulation," *Advanced Packaging, IEEE Transactions on* , vol.30, no.4, pp.763-775, Nov. 2007

[43] S. Lefteriu and A. C. Antoulas, "A new approach to modeling multiport systems from frequency-domain data," *IEEE Tran. Comp. Aided Des. Integ. Cir. Sys.*, vol. 29, no. 1, pp. 14–27, 2010.

[44] S. Lefteriu, A. C. Antoulas, and A. C. Ionita, "Parametric model order reduction from measurements," *IEEE Conf. on Electrical Performance of Electronic Packaging and Systems (EPEPS)*, pp.193–196, 2010.

[45]A.C. Antoulas, A.C. Ionita, S. Lefteriu," On two-variable rational interpolation", *Lin. Alg. App.*, Vol. 436, no. 8, pp. 2889–2915, 2012.

[46] Y. Wang, C. Lei, G.K.H. Pang, and N. Wong, "MFTI: Matrix-Format Tangential Interpolation for Modeling Multi-Port Systems", in *Proc. ACM/IEEE conf. on DAC*, pp. 683–686, July 2010.

[47] M. Kabir and R. Khazaka, “Macromodeling of Distributed Networks From Frequency-Domain Data Using the Loewner Matrix Approach,” *IEEE Tran. on Microwave Theory and Techniques*, Vol. 60, no. 12, pp. 3927–3938, Dec. 2012.

[48] M. Kabir and R. Khazaka, “Parametric Macromodeling of High-Speed Modules from Frequency-Domain Data using Loewner Matrix based Method”,

[49] Hiai, Fumio; Sano, Takashi,” Loewner matrices of matrix convex and monotone functions”, eprint arXiv:1007.2478, 07/2010

[50] Kabir, M.; Khazaka, R.; Achar, R.; Nakhla, M., "Loewner-Matrix based efficient algorithm for frequency sweep of high-speed modules," *Electrical Performance of Electronic Packaging and Systems (EPEPS), 2012 IEEE 21st Conference on* , vol., no., pp.185,188, 21-24 Oct. 2012

[51] Coifman, R.; Rokhlin, V.; Wandzura, S.; , "The fast multipole method for the wave equation: a pedestrian prescription," *Antennas and Propagation Magazine, IEEE* , vol.35, no.3, pp.7-12, June 1993

[52] Odabasioglu, A.; Celik, M.; Pileggi, L.T., "PRIMA: passive reduced-order interconnect macromodeling algorithm," *Computer-Aided Design, 1997. Digest of Technical Papers., 1997 IEEE/ACM International Conference on* , vol., no., pp.58,65, 9-13 Nov. 1997

[53] Lefteriu, S.; Antoulas, A.C., "Modeling multi-port systems from frequency response data via tangential interpolation," *Signal Propagation on Interconnects, 2009. SPI '09. IEEE Workshop on* , vol., no., pp.1,4, 12-15 May 2009

[54]Sanda Lefteriu, “New Approaches to Modeling Multi-Port Scattering Parameters”, Master thesis, HOUSTON, TEXAS, JUNE 2008

[55]Z. Liu, M. Kabir and R. Khazaka, “Time-Domain SPICE Macromodel of High-Speed Modules from Y -parameter Data using Loewner Matrix Approach,” *2013 International Conference on Analog VLSI Circuits (AVIC 2013),* Proceeding, pp. 90–93, Oct. 2013

[56] E. A. Guillemin, *Synthesis of Passive Networks*. John Wiley & Sons,Inc., 1957.

[57] Brune, O.: Synthesis of a Finite Two-Terminal Network Whose Driving-Point Impedance is a Prescribed Function of Frequency, J. Math. Phys. Camb., 10, 191–236, 1931.

[58]R.M. Foster, "A Reactance Theorem," Bell Syst. Tech. J., vol. 3, no. 2, pp. 259-267, April 1924.

[59]H.J. Carlin and D.C. Youla, "Network Synthesis with Negative Resistors," Proc. IRE, vol. 49, no. 5, pp. 907-920, May 1961.

[60] P. L. Werner, R. Mittra, and D. H. Werner, “Extraction of SPICE-type equivalent circuits of microwave components and discontinuities using the genetic algorithm optimization technique,” *IEEE Trans. Adv. Packag.*, vol. 23, pp. 55–61, Feb. 2000.

[61]Caratelli, D.; Haider, N.; Yarovoy, A., "Analytically based extraction of Foster-like frequency-independent antenna equivalent circuits," *Electromagnetic Theory (EMTS), Proceedings of 2013 URSI International Symposium on* , vol., no., pp.506,509, 20-24 May 2013

[62] F. Mukhtar, Y. Kuznetsov, and P. Russer, “Network modelling with brune’s synthesis,” in *Kleinheubacher Tagung*, Oct. 2010, kH2010-B-1493.

[63] V. Belevitch, “On the brune’s process for n-ports,” *IRE Transactions on Circuit Theory*, vol. 7, no. 3, pp. 280–296, September 1960.

[64] B. D. H. Tellegen, “Synthesis of the 2n-poles by networks containing the minimum number of elements,” *Journal of Mathematics And Physics*, vol. 32, no. 1, pp. 1–18, April 1953.

[65]Mukhtar, F.; Kuznetsov, Y.; Hoffmann, C.; Russer, P., "Brune's synthesis of linear lossy distributed one-port and symmetric two-port microwave circuits," *Microwave Conference (GeMIC), 2011 German* , vol., no., pp.1,4, 14-16 March 2011

[66]Caratelli, D.; Haider, N.; Yarovoy, A., "Analytically based extraction of Foster-like frequency-independent antenna equivalent circuits," *Electromagnetic Theory (EMTS), Proceedings of 2013 URSI International Symposium on* , vol., no., pp.506,509, 20-24 May 2013

[67] Yidi Song, *Parallel Vector Fitting of Systems Characterised by Measured or Simulated Data,* Master thesis, McGill University, Montreal, 2013

[68]Anton, Howard; Rorres, Chris (2005). *Elementary Linear Algebra* (9th ed.). John Wiley and Sons, Inc. ISBN 978-0-471-66959-3.

[69] Roger A. Horn and Charles R. Johnson (1991), *Topics in matrix analysis,* Cambridge University Press. *Section 6.1*

[70] CHEN Michael Z. Q. ; SMITH Malcolm C., *A Note on Tests for Positive-Real Functions,* IEEE transactions on automatic control Y. 2009, vol. 54, No. 2, pages 390-393. ISSN : 0018-9286

[71] Sarto, M.S.; Scarlatti, A; Holloway, C.L., "On the use of fitting models for the time-domain analysis of problems with frequency-dependent parameters," *Electromagnetic Compatibility, 2001. EMC. 2001 IEEE International Symposium on* , vol.1, no., pp.588,593 vol.1, 2001

[72] Antonini, Giulio. "SPICE equivalent circuits of frequency-domain responses." *Electromagnetic Compatibility, IEEE Transactions on* 45.3 (2003): 502-512.

[73] Green G, "An Essay on the Application of Mathematical Analysis to the Theories of Electricity and Magnetism" (Nottingham, England: T. Wheelhouse, 1828). pages 10-12

[74] Bjørn Gustavsen, The Vector Fitting Web Site, retrieved from https://www.sintef.no/projectweb/vectfit/downloads/