# Model Order Reduction- VLSI Ckts

Vineet Sahula

sahula@ieee.org

1.2 *RLC* Interconnect Circuit Formulation 18
1.3 General *RLC* Interconnect Circuit Formulation 20
1.4 Remarks 23
2 Responses in Time Domain 24
2.1 Responses in Closed-Form 24
2.2 Taylor Expansion in Time Domain 24
3 $s$ Domain Analysis 27
3.1 Transfer Function 27
3.2 Responses from $s$ Domain to Time Domain 28
4 Preliminaries of Symbolic Analysis 29
4.1 Matrix, Determinant, and Cofactors 29
4.2 Cramer's Rule 30

Part II Linear VLSI Circuits

3. MODEL-ORDER REDUCTION 35
1 $s$ Domain Analysis 35
2 Moments and Moment-Matching Method 35
2.1 Concept of Moments 35
2.2 Delay Estimation Using Moments 36
2.3 Deriving Moments from MNA Formulation 41
2.4 Deriving Moments for RLC Trees 41
3 Realizable Topological Reduction Methods 46
3.1 TICER 48
3.2 Realizable RLC $\pi$-Model Reduction 51
3.3 Scattering-Parameter-Based Macro Model Reduction 54
4 Summary 61

4. GENERALIZED Y-Δ TRANSFORMATION — FUNDAMENTAL THEORY 63
1 Introduction 64
2 Classical Y-Δ Transformation 65
2.1 Numerical Example 65
2.2 Y-Δ Transformation and Gauss Elimination 66
2.3 Notations and Terminologies to be Used 67
3 Generalized Y-Δ Reduction 67
3.1 Branch with RCLK Elements 68
3.2 Branches with Current and Voltage Sources 70



3.3 RCLK-VJ Generalized Formulae for Y-Δ Transformation 71
3.4 Higher-Order Truncation 72
4 Node Ordering 73
5 Generalized Y-Δ Reduction Flow 74
6 Summary 76

5. GENERALIZED Y-Δ TRANSFORMATION — ADVANCE TOPICS 77
1 Common-Factor Effects 77
1.1 Example on Common-Factor Effects 77
1.2 Existence of Common Factors 79
1.3 Common Factors in Current Source Transformation 82
2 Revised Generalized Y-Δ Reduction Flow — A Redundancy-Free Version 83
3 Multiport Y-Δ Reduction 85
3.1 Backward-Solving in LU Factorization 85
3.2 Δ-Y Recovery 87
3.3 Multiport Y-Δ Reduction Flow 89
4 Treating Roundoff Errors 89
4.1 Fundamentals of Roundoff Errors 89
4.2 Roundoff Errors in Y-Δ Transformations 91
4.3 Solution to Roundoff Problems in Y-Δ Transformation 92
5 Experimental Results 95
6 Summary 97
7 Appendices 97
7.1 Existence of Type-II Common Factor 97
7.2 Recursive Existence of Type-II Common Factor 103
7.3 Existence of Type-I and Type-II Common Factors in Current Source Transformation 103
7.4 Simplest Form of Y-Δ Transformation 111

6. Y-Δ TRANSFORMATION: APPLICATION I — MODEL STABILIZATION 117
1 Hurwitz Polynomial 117
2 The Routh-Hurwitz Criterion 118
2.1 Necessary Conditions 118
2.2 The Routh-Hurwitz Criterion — A Necessary and Sufficient Condition 119
2.3 Proof of the Routh-Hurwitz Criterion 123

3 Stabilizing Models After Y-Δ Reduction 136
4 Experimental Results 140
5 Summary 140

7. Y-Δ TRANSFORMATION: APPLICATION II — REALIZABLE PARASITIC REDUCTION 141
1 First-Order realization 141
2 Admittance Not Realizable in Nature 142
3 Idea of Templates 143
4 Geometric Programming 144
4.1 Intuitions 144
4.2 Primal and Dual Functions 145
4.3 Orthogonality Conditions 146
4.4 Solution Space of Dual Problems 146
4.5 Geometric Programming with Constraints 146
5 Template Realization Using Geometric Programming 149
6 Experimental Results 150
7 Summary 152

Part III Analog VLSI Circuits

8. TOPOLOGICAL ANALYSIS OF PASSIVE NETWORKS 155
1 Review of Node Admittance Matrix 155
1.1 Incidence Matrix of Undirected Graph 155
1.2 Incidence Matrix of Directed Graph 156
1.3 Composition of Node Admittance Matrix 159
2 Problem Formulation 160
2.1 Driving-Point Admittance $Y_i(s)$ 161
2.2 Open-Circuit Impedance $Z_{ij}(s)$ 162
2.3 Network Transfer Functions 164
3 Topological Formulas 164
3.1 Topological Formula for Determinant Δ 165
3.2 Topological Formula for $\Delta_{ij}$ 168
3.3 Time Complexity 173
4 Flow-Graph Technique 174
5 Summary 183



9. EXACT SYMBOLIC ANALYSIS USING DETERMINANT DECISION DIAGRAMS 185
1 Combination Set Systems and Zero-Suppressed Binary Decision Diagrams 185
2 DDD Representation of Symbolic Matrix Determinant 187
3 An Effective Vertex Ordering Heuristic 191
4 Manipulation and Construction of DDD Graphs 196
4.1 Implementation of Basic Operations 197
4.2 Illustration of Basic Operations and its Use in Circuit Sensitivity 200
5 DDD-based Exact Symbolic Analysis Flow 203
6 Related to Other Decision Diagram Concepts 204
7 Application to Symbolic Analysis of Analog Circuits 206
8 Summary 208
9 Historical Notes on Symbolic Analysis Techniques 209

10. S-EXPANDED DETERMINANT DECISION DIAGRAMS FOR SYMBOLIC ANALYSIS 211
1 Introduction 211
2 s-Expanded Symbolic Representations 212
3 Vertex Ordering for s-Expanded DDDs 215
4 Construction of s-Expanded DDDs 217
4.1 The Construction Algorithm 217
4.2 Time and Space Complexity Analysis 218
5 Applications of Deriving Transfer Functions 221
6 Summary 222

11. DDD BASED APPROXIMATION FOR ANALOG BEHAVIORAL MODELING 225
1 Introduction 225
2 Symbolic Cancellation and De-cancellation 226
3 Dynamic Programming based Generation of Dominant Terms 228
4 Incremental k-Shortest Path Algorithm 231
4.1 DDD-based Approximation Flow 237
5 Application to AC Characterization of Analog Circuits 238
6 Summary 240
7 Historical Notes on Symbolic Approximation 240
8 Appendix 241

12. HIERARCHICAL SYMBOLIC ANALYSIS AND HIERARCHICAL MODEL ORDER REDUCTION 245
1 DDD-based Hierarchical Decomposition 246
1.1 Subcircuit Reduction 246
1.2 Overview of The Simulation and Reduction Algorithm 249
2 DDD-based Hierarchical Decomposition 250
3 Cancellation Analysis for Subcircuit Reduction 253
3.1 Cancellation Due to Circuit Devices 253
3.2 Cancellation Due to Subcircuit Reduction 254
3.3 Theoretical Analysis of Cancellation Conditions 257
3.4 Device-Level Cancellation From Subcircuit Reduction's Perspective 259
3.5 Cancellation at Different Hierarchical Circuit Levels 259
4 General s-Domain Hierarchical Network Modeling and Simulation Algorithm 261
4.1 Cancellation-Free Rational Admittance 261
4.2 Y-expanded DDDs 262
4.3 Computation of Cancellation-Free Rational Admittances 264
4.4 Clustering Algorithm 267
5 Hierarchical Analysis of Analog Circuits – Examples 267
6 Summary 270
7 Historical Notes on the Model Order Reduction 271
References 273
Index 281

# List of Figures

1.1 Symbolic analysis of a simple RC circuit 4
1.2 RC circuit for illustration of DDD 10
1.3 A matrix determinant and its DDD representation. 11
2.1 A $RC$ tree demonstrating nodal analysis formulation: current flowing out of a node is equal to currents flowing into the node 16
2.2 $RLC$ parasitics of a segment of interconnect on metal layers:(a) illustration of orthogonal interconnect layers (b) $RLC$ parasitic model of an interconnect segment in layer 3. 18
2.3 $RLC$ circuit meeting the two prerequisites. Shaded ones are the so-called intra-branch nodes 19
2.4 A RLC tree demonstrating modified nodal analysis formulation. 22
2.5 A linear network with two inputs and three outputs. 28
3.1 Scenario that the median and mean points mismatch: when the unit impulse response $H(t)$ (scaled) is not symmetric, 50% delay of unit step response $Y(t)$, or the median point of $H(t)$, does not overlap with the mean point of $H(t)$. 37
3.2 Scenario that the median and mean points of $H(t)$ overlap: when $H(t)$ is symmetric, 50% delay point of $Y(t)$ matches the mean point of $H(t)$, i. e., the approximation is exact. 38
3.3 The original RLC tree in Example 3.3. 43
3.4 The "resistive tree" for computing $M_0$ in Example 3.3. 45
3.5 The "resistive tree" modified based on the original RLC circuit by zeroing out inputs and replacing capacitors and inductors with current and voltage sources, respectively. 46

## 1.1 A "Hello-World" Example

As a simple example, Fig. 1.1 shows a circuit with two resistors and two capacitors. This circuit is stimulated by a voltage source, and the other end is

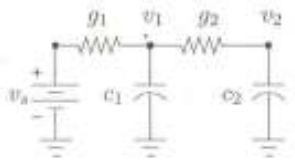



*Figure 1.1.* Symbolic analysis of a simple RC circuit

grounded. *Symbolically* in $s$ domain, $v_1$ can be written as

$$v_1 = \frac{g_1 g_2}{g_1 g_2 + (g_1 c_2 + g_2 c_2 + c_1 g_2)s + c_1 c_2 s^2} v_s.$$

This symbolic equation tells us that it is a second order low-pass filter with DC gain = 1. Of course for this simple circuit, every electrical engineer can write up the actual equation in the old pencil-and-paper style. But when circuits become large, such kind of transfer functions can grow exponentially in terms of number of product terms such as $g_1 g_2$. On the other hand, symbolic analysis may be able to generate them automatically.

If a symbolic analyzer produces a symbolic transfer function with hundreds or thousands of symbolic product terms and with all circuit parameters mixed together, it's hard to identify the influence of every parameter to the whole function. Simplification can be applied to sort out the dominant terms of parameters. There are two strategies of doing this:

- Mixed symbolic-algebraic analysis keeps only a small number of circuit parameters as symbols and the rest as numerical values. It helps reduce the number of total product terms and the length of each of them. The extreme of this approach is the algebraic analysis in which *all* circuit parameters are numerical values and the only symbol in the expressions is the complex frequency $s$.
- Symbolic simplification discards insignificant terms based on the relative magnitudes of symbolic parameters and the frequency defined at some nominal design points or over some ranges. It can be performed before, during, or after the generation of symbolic terms [8, 42, 73, 107, 27, 37, 98].

## 1.2 Problem Formulation for Symbolic Analysis

Consider a lumped linear(ized) time-invariant analog circuit in frequency domain. Its circuit equation can be formulated, for example, by the nodal



analysis approach in the following general form [96]:

$$Ax = b. \tag{1.1}$$

The *circuit unknown vector* $x$ may be composed of $n$ node voltages, and the *admittance matrix* $A$ is an $n \times n$ sparse symbolic matrix, $b$ is a vector of external sources.

Symbolic analysis of analog circuits can be stated as the problem of solving the symbolic equation (1.1), i. e., to find a symbolic expression of any circuit unknowns in terms of symbolic parameters in $A$ and symbolic excitations expressed by $b$. According to Cramer's rule, the $k$th component $x_k$ of the unknown vector $x$ is obtained as follows:

$$x_k = \frac{\sum_{i=1}^{n} b_i \, (-1)^{i+k} \, det(A_{a_{i,k}})}{det(A)}, \tag{1.2}$$

where $det(A)$ is the determinant of matrix $A$, and $(-1)^{i+k} det(A_{a_{i,k}})$ in (1.2) is the cofactor of $det(A)$ with respect to element $a_{i,k}$ of matrix $A$ at row $i$ and column $k$.

Most symbolic simulators are targeted at finding various network functions, each being defined as the ratio of an output from $x$ to an input from $b$. Generally, a transfer function of a linear(ized) circuit can be obtained as a rational function in the complex frequency variable $s$:

$$H(s) = \frac{\sum f_i(p_1, p_2, \cdots, p_m) s^i}{\sum g_j(p_1, p_2, \cdots, p_m) s^j}, \tag{1.3}$$

where $f_i(p_1, p_2, \cdots, p_m)$ and $g_j(p_1, p_2, \cdots, p_m)$ are symbolic polynomial functions in circuit parameters $p_j, j = 1, ..., m$. These polynomials in turn can be expressed in a nested form or an expanded sum-of-product form.

Again, we can categorize symbolic analysis in terms of number of symbols in a given circuit:

1. If the polynomial coefficients, $f_i(\cdots)$ and $g_j(\cdots)$, contain only symbols, it is named *fully* or *exact* symbolic analysis.
2. If only some circuit parameters are represented as symbols, it is named *partial* or *mixed* symbolic analysis.
3. In the extreme case is that transfer function $H(s)$ has only one symbol – the complex frequency $s$, which happens when all circuit parameters are numerical values and the symbolic analysis degenerates to algebraic analysis.

The central issue in symbolic analysis is to find symbolic expressions of $det(A)$ and the cofactors of $det(A)$.

## 2. Linear Circuit Reduction

linear circuits targeted in this book refer to interconnect parasitics in modern VLSI designs. Interconnects become more important to digital circuit designers than never before, because the quality of the interconnects needs to be examined in every aspect from delay to signal integrity issues.

Although accurate analysis is preferable, turn-around time is never a negligible factor to be considered. Because a huge amount of interconnects are present in typical VLSI designs. Nowadays, a million-transistor design can easily accommodate miles of interconnects.

To analyze interconnect parasitics accurately and quickly, linear circuit reduction has to be adopted. Researchers have devised a lot of techniques on this hot topic. Their works can be classified into two categories:

- projection-based model order reduction.
- generalized Y-$\Delta$ transformation.

### 2.1 Projection-Based Model Order Reduction

Modeling complicated linear circuits with simple yet accurate circuits is called *model order reduction*. The idea is to analyze or simulate the simplified models to reduce circuit verification time.

A number of projection-based model-order reduction based techniques have been introduced [23, 24, 25, 64, 66, 80, 79] to analyze the transient behavior of interconnects. Those projection-based algorithms mainly work for passive linear networks as the computation of moments and Krylov space base vectors requires a special partitioning of circuit matrices and solving of the partitioned circuit matrices iteratively.

The reduction is typically done in frequency or $s$ domain, where $s$ is the complex frequency variable and defined as $j\omega$, where $\omega$ is the radical frequency in complex domain analysis. In frequency domain analysis, storage elements such as capacitors and conductors all have their impedance written in $s$ in frequency domain. For example, a capacitor with capacitance value 0.1 farad can be written as $0.1s$ in frequency domain as its impedance value.

### 2.2 Generalized Y-$\Delta$ Transformation

Another different approach to circuit complexity reduction is by means of local node reduction and realization of reduced networks based on local node elimination and realization [22, 74, 2, 69, 75, 82]. The main idea is to reduce the number of nodes in the circuits and approximate the elements of the reduced system with either order-reduced rational functions or realized low order $RLCM$ networks. The major advantage of these methods over projection-based methods is that the reduction can be done in a local manner and no overall solutions



of the entire circuit are required, which makes these methods very amenable to attack large linear networks. This idea was first explored by selective node elimination for RC circuits [22, 74], where time constant analysis is used to select nodes for elimination.

Generalized Y-$\Delta$ transformation [69, 68], $RLCK$ circuit crunching [2], and branch merging [75] have been developed based on nodal analysis, where inductance becomes susceptance in the admittance matrix. Generalized Y-$\Delta$ transformation provides a general node elimination based parasitic reduction technique [68].

A generalized block Y-$\Delta$ transformation based on modified nodal analysis formulation has been proposed [82, 83] recently, which leads to the general hierarchical model order reduction techniques and it can be applied to any linear circuits with any linear device. Both Generalized Y-$\Delta$ transformation and hierarchical model order reduction techniques will be discussed in detail in the following chapters.

Both projection-based and node elimination based model order reduction methods can be viewed a special symbolic analysis where only the complex frequency variable is the symbol. In general, transfer functions are functions of $s$, which are called semi-symbolic analysis format from the perspective of symbolic analysis.

## 3. Symbolic Analysis for Analog Circuit in a Nutshell

Research on symbolic analysis can be dated back to the last century. Developments in this field gained real momentum in 1950's when electric computers were introduced and used in circuit analysis. Methods developed from the 1950's to the 1980's can be basically categorized as:

1. *Tree Enumeration methods,*
2. *Signal Flow Graph methods,*
3. *Parameter Extraction methods,*
4. *Numerical Interpolation methods* and
5. *Matrix-Determinant methods.*

The details of these method can be found in [38, 53]. In the late 1980's, symbolic analysis gains renewed interests as industrial demands for analog design automation increased. Various methods are proposed to solve the long-standing circuit-size problem. The strategies used in modern symbolic analyzers in general come in two categories: those based on hierarchical decompositions [41, 81, 91] and those based on approximations [30, 98, 107, 42, 107, 18, 48, 49, 86].

approximation, passive reduction, and realizable reduction, with each more sophisticated than the previous one.

## 1. Time Domain Analysis

In this section, we describe $RC$ and $RLC$ network analysis in time domain progressively in three steps. Our first step is to present an approach to analyzing $RC$ circuits. And then the second step is to formulate $RLC$ circuits with certain special structure. Our last step is to introduce the formulation of $RLC$ circuits of general structure. Each of them begins with a simple example and are presented formally in matrix terminologies afterwards.

### 1.1 *RC* Interconnect Circuit Formulation

$RC$ interconnect circuits can be formulated using nodal analysis formulation. Nodal analysis is a classical circuit analysis method based on Kirchhoff's Current Law[1] (KCL) and branch constitutive equations[2]. For a given $RC$ linear circuit with $n+1$ nodes, nodal analysis formulates the problem in the following two steps:

**step 1.** choose a ground or reference node, which usually is taken to be at a potential of zero volt. All other node voltages constitute $n$ unknowns[3];

**step 2.** establish KCL equations for all the $n$ nodes by representing branch currents in terms of node voltages using branch constitutive equations.

**Example 2.1.** Refer to the $RC$ tree in Fig. 2.1. In nodal analysis formulation of the circuit, we first determine the unknowns. Since $v_1$ is equal to $v_s$ which

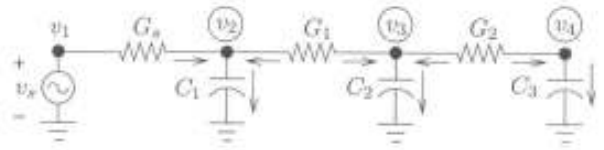



*Figure 2.1.* A $RC$ tree demonstrating nodal analysis formulation: current flowing out of a node is equal to currents flowing into the node

is the given input, we use node voltages $v_2$, $v_3$, and $v_4$ as unknown variables to

[1]Kirchhoff's Current Law: for lumped circuits, the algebraic sum of the currents entering (leaving) a node is zero.

[2]A branch constitutive equations are $i$-$v$ relationships for circuit elements such as resistors, capacitors, inductors, dependent and controlled sources, etc. For example, the branch constitutive equation for a resistor of value $r$ is $i = v/R$.

[3]The $n$ node voltages are independent because they linearly represent $n$ independent voltage drops on tree trunks on any tree of the circuit.



write the three KCL equations

$$C_1\dot{v}_2 = G_s(v_s - v_2) + G_1(v_3 - v_2) \tag{1.1}$$
$$C_2\dot{v}_3 = G_1(v_2 - v_3) + G_2(v_4 - v_3) \tag{1.2}$$
$$C_3\dot{v}_4 = G_2(v_3 - v_4), \tag{1.3}$$

or if we order the unknown variables on the right-hand side of the equations, we may have

$$C_1\dot{v}_2 = -(G_s + G_1)v_2 + G_1v_3 + G_sv_s \tag{1.4}$$
$$C_2\dot{v}_3 = G_1v_2 - (G_1 + G_2)v_3 + G_2v_4 \tag{1.5}$$
$$C_3\dot{v}_4 = G_2v_3 - G_2v_4 \tag{1.6}$$

The matrix form of the above three simultaneous equations would be

$$\begin{bmatrix} C_1 & & \\ & C_2 & \\ & & C_3 \end{bmatrix} \begin{bmatrix} \dot{v}_2 \\ \dot{v}_3 \\ \dot{v}_4 \end{bmatrix} = - \begin{bmatrix} G_s + G_1 & -G_1 & 0 \\ -G_1 & G_1 + G_2 & -G_2 \\ 0 & -G_2 & G_2 \end{bmatrix} \begin{bmatrix} v_2 \\ v_3 \\ v_4 \end{bmatrix} + \begin{bmatrix} G_sv_s \\ 0 \\ 0 \end{bmatrix}. \tag{1.7}$$

Note that we have put a minus sign outside the square matrix.

Without loss of generality, we assume that the voltage drop and the current from ② to ③ are two output variables that we are interested,i. e., $i_x$ ans $v_x$ are circuit outputs:

$$\begin{bmatrix} i_x \\ v_x \end{bmatrix} = \begin{bmatrix} 1 & -1 & 0 \\ G_1 & -G_1 & 0 \end{bmatrix} \begin{bmatrix} v_2 \\ v_3 \\ v_4 \end{bmatrix}. \tag{1.8}$$

As the reader can imagine, we can use some linear combination of the unknowns to obtain voltages between any two nodes or currents on any branch in the circuit.

In (1.7), each row is derived from KCL for each node in Fig. 2.1. For example, the first equation states that the current flowing out of node ② through $C_1$, i. e., $C_1\dot{v}_1$, is equal to the currents flowing into the node through $G_s$ and $G_1$, which are $G_s(v_s - v_2)$ and $G_1(v_3 - v_2)$, respectively.

In general, $RC$ circuit formulation can be expressed as

$$C\dot{V}(t) = -GV(t) + PU(t) \tag{1.9}$$
$$Y(t) = QV(t) \tag{1.10}$$

where $V$ denotes the $n$ unknown nodal voltages in $RC$ circuits. In (1.9), the matrices $G \in \Re^{n\times n}$ and $C \in \Re^{n\times n}$ represent the conductance and capacitance elements, respectively.

Please note that matrix $C$ may be singular, i. e., some rows in $C$ may be zero. It happens when the corresponding node does not connect to any capacitor. It is

worth noting as well that the $G$ matrix in (1.7) is non-singular if and only if the $RC$ interconnect circuits meet the requirement that each node has one or more resistive path to some other nodes. Being a non-singular matrix is a necessary condition for the most linear reduction techniques.

## 1.2 *RLC* Interconnect Circuit Formulation

Formulation of a type of $RLC$ circuits can be easily obtained by augmenting the $RC$ formulation we have introduced. The $RLC$ circuits in this class require that every inductor must be in series with a resistor. In fact when $L$ is considered, the parasitics of an interconnect segment typically is modeled as a $RLC$ branch as shown in Fig. 2.2. The branch constitutive equation of $L$ in Fig. 2.2 is given

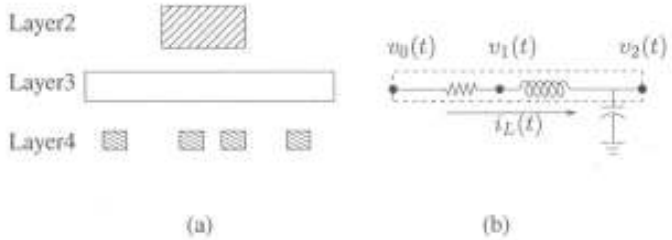



*Figure 2.2.* $RLC$ parasitics of a segment of interconnect on metal layers:(a) illustration of orthogonal interconnect layers (b) $RLC$ parasitic model of an interconnect segment in layer 3.

by

$$v_1(t) - v_2(t) = L\dot{i}_L(t), \tag{1.11}$$

where $v_1(t)$ and $v_2(t)$ are the two nodal voltages of the inductor.

The formulation method that we described for $RC$ circuits can be used to formulate $RLC$ circuits with minor modifications. We have known that each row in the formulation is constituted by KCL. KCL equations are established in terms of nodal voltages and their derivatives as unknowns. If we want to keep the formulation, $i_L(t)$ has to be represented with nodal voltages.

In general, $i_L(t)$ can be calculated by

$$i_L(t) = \frac{1}{L}\int_{t_0}^{t}(v_1(t) - v_2(t))\mathrm{d}t + i_L(t_0), \tag{1.12}$$

This integral equation, however, is apparently not a fit to our $RC$ formulations because only nodal voltages and their derivatives can be used as unknowns.

Fortunately, provided that the inductor is in series with a resistor in our $RLC$ circuits, $i_L(t)$ is equal to the current flowing through the resistor as well, i.e.,

$$i_L(t) = G\left(v_0(t) - v_1(t)\right), \tag{1.13}$$



where $v_0(t)$ and $v_1(t)$ are the two nodal voltages of the resistor. Therefore, insert (1.13) into (1.11), we can rewrite (1.11) into the form of

$$\begin{aligned} v_1(t) - v_2(t) &= L\dot{i}_L(t) \\ &= LG\left(\dot{v}_0(t) - \dot{v}_1(t)\right). \end{aligned} \tag{1.14}$$

Essentially we have used the nodal voltages of the resistor to represent the current of the inductor.

$RLC$ circuit formulation can be augmented based on $RC$ formulation as follows:

**step 1.** choose a ground or reference node, which usually is taken to be at a potential of zero volt. All other node voltages constitute $n$ unknowns;

**step 2.** establish KCL equations for all the *inter-branch nodes (those on the joints of branches)* by representing branch currents in terms of node voltages using branch constitutive equations. *For branch that is an inductor, use (1.13) to represent the current in the inductor.*

**step 3.** establish (1.14) for all intra-branch nodes (those inside branches between resistors and inductors).

**Example 2.2.** We change the circuit in Fig. 2.1 to the one in Fig. 2.3 by adding two inductors $L_1$ and $L_2$ in series with $G_1$ and $G_2$, respectively. This circuit structure meets our requirement: $L_1$ is in series with $G_1$, and $L_2$ is in series with $G_2$.

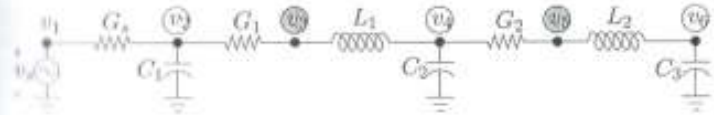



*Figure 2.3.* $RLC$ circuit meeting the two prerequisites. Shaded ones are the so-called intra-branch nodes

The circuit can be formulated as (1.15). The first three equations are established based on Step 2. While the last two are based on Step 3. In terms of nodes, the first three rows are derived from KCL for three inter-branch nodes. The last two rows are modified branch constitutive equations (1.14) for two

intra-branch nodes.

$$\begin{bmatrix} C_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & C_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & C_3 \\ -G_1L_1 & G_1L_1 & 0 & 0 & 0 \\ 0 & 0 & -G_2L_2 & G_2L_2 & 0 \end{bmatrix} \begin{bmatrix} \dot{v}_2 \\ \dot{v}_3 \\ \dot{v}_4 \\ \dot{v}_5 \\ \dot{v}_6 \end{bmatrix} =$$
$$- \begin{bmatrix} G_s+G_1 & -G_1 & 0 & 0 & 0 \\ -G_1 & G_1 & G_2 & -G_2 & 0 \\ 0 & 0 & -G_2 & G_2 & 0 \\ 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{bmatrix} + \begin{bmatrix} G_s v_s \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \quad (1.15)$$

Letting the branch current and voltage of $L_2$ be the outputs, we have

$$\begin{bmatrix} v_{L_2} \\ i_{L_2} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & G_2 & -G_2 & 0 \end{bmatrix} \begin{bmatrix} v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{bmatrix}. \quad (1.16)$$

In general, $RLC$ circuit formulation can be expressed as

$$C\dot{V}(t) = -GV(t) + PU(t) \quad (1.17)$$
$$Y(t) = QV(t) \quad (1.18)$$

Similar to the $RC$ formulation, $V$ denotes the $n$ unknown nodal voltages in $RLC$ circuits, and matrices $G \in \Re^{n\times n}$ and $C \in \Re^{n\times n}$ represent the conductance and capacitance elements. In addition, the two matrices contain other elements such as 1 and $GL$ due to the introduction of modified branch constitutive equations (1.14).

Same as the $RC$ formulation, matrix $C$ may be singular. We assume that this will not happen in $RLC$ circuits throughout our discussion. The same assumption for $RC$ circuits applies to the $RLC$ circuits to ensure that matrix $G$ is non-singular.

## 1.3 General $RLC$ Interconnect Circuit Formulation

For more general $RLC$ circuits of which our assumption to the topology of $L$ does not hold, a more general formulation is needed. Modified nodal analysis is yet another classical circuit formulation method which improves nodal analysis method by adding currents in inductors as unknown variables. The introduction of the inductance current variables would help keep modified nodal analysis formulation in the differential form.



For example, the branch constitutive equation of an inductor is

$$v_L = L\frac{\mathrm{d}i_L}{\mathrm{d}t}, \quad (1.19)$$

where $L$ is the inductance value. If we had to use nodal analysis, in the KCL equations involving the inductor, $i_L$ has to be represented in terms of $v_L$, i. e., $i_L = \frac{1}{L}\int_{t_0}^{t} v_L \mathrm{d}t + i(t_0)$. While in the modified version, introducing $i_L$ into the unknowns would keep the KCL equations in the differential form. The cost, however, is an additional equation (1.19).

Our general $RLC$ circuit formulation can take one step further from modified nodal analysis, additional inductance current variables can be removed from the formulation by block Gauss elimination. However, this can be done only in $s$ domain.

**step 1.** choose a ground or reference node, which usually is taken to be at a potential of zero volt. All other node voltages constitute $n$ unknowns;

**step 2.** create a current variable for each inductor with certain direction defined;

**step 3.** establish KCL equations for all the $n$ nodes by representing branch currents of $RC$ elements in terms of node voltages and current variables pre-defined in step 2;

**step 4.** establish the branch constitutive equation of inductance in differential form of (1.19) using pre-defined current and nodal voltage variables;

**step 5 (optional).** remove current variables using block Gauss elimination in $s$ domain.

Step 1-4 are the procedure of modified nodal analysis on general $RLC$ circuits. Step 5 is the post-procedure for removal of current variables.

**Example 2.3.** Fig. 2.4 shows an $RLC$ circuit with a mutual inductance $M$ between $L_1$ and $L_2$. Note that $L_2$ in the circuit does not meet our assumption in 1.2, i. e., it does not run in series with any resistor.

In order to formulate the circuit using modified nodal analysis, $i_1$ and $i_2$ are two current variables in addition to the four nodal voltages. The modified nodal

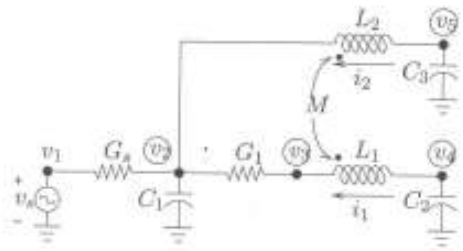



*Figure 2.4.* A RLC tree demonstrating modifi ed nodal analysis formulation.

analysis formulation of the circuit is given by

$$\left[\begin{array}{cccc|cc} C_1 & & & & 0 & 0 \\ & 0 & & & 0 & 0 \\ & & C_2 & & 0 & 0 \\ & & & C_3 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & L_1 & M \\ 0 & 0 & 0 & 0 & M & L_2 \end{array}\right] \left[\begin{array}{c} \dot{v}_2 \\ \dot{v}_3 \\ \dot{v}_4 \\ \dot{v}_5 \\ \dot{i}_1 \\ \dot{i}_2 \end{array}\right] =$$

$$- \left[\begin{array}{cccc|cc} G_s+G_1 & -G_1 & 0 & 0 & 0 & -1 \\ -G_1 & G_1 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 1 & -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & -1 & 0 & 0 \end{array}\right] \left[\begin{array}{c} v_2 \\ v_3 \\ v_4 \\ v_5 \\ i_1 \\ i_2 \end{array}\right] + \left[\begin{array}{c} G_s v_s \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{array}\right]. \quad (1.20)$$

Let $v_6$ be the output voltage; then we have

$$\left[v_6\right] = \left[\begin{array}{cccccccc} 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right] \left[\begin{array}{c} v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ i_1 \\ i_2 \end{array}\right]. \quad (1.21)$$

In (1.20), the first four rows are derived from KCL for the five circled nodes. The last two rows are branch constitutive equations of the two inductors and the mutual one, which are added because of the two extra variables, $i_1$ and $i_2$.

In general, modified nodal analysis formulation can be expressed as

$$M\dot{x}(t) = -Gx(t) + Pu(t) \quad (1.22)$$

$$y(t) = Qx(t) \quad (1.23)$$



where

$$x(t) \equiv \begin{bmatrix} V(t) \\ I(t) \end{bmatrix} \quad M \equiv \begin{bmatrix} C & 0 \\ 0 & L \end{bmatrix} \quad G \equiv \begin{bmatrix} G_1 & W^T \\ E & 0 \end{bmatrix} \quad (1.24)$$

where $V$ and $I$ are the modified nodal analysis variables (yielding a total number of $n$ unknowns in (1.22)) corresponding to the node voltages and the branch currents for floating voltage sources and inductors. In (1.22), the matrices $G \in \Re^{n\times n}$ and $M \in \Re^{n\times n}$ represent the conductance and susceptance matrices (except that the rows corresponding to the current variables are negated). In (1.24), $C$ and $L$ are generally capacitance and inductance matrices of the circuit. However, please note that $C$ may be singular, i. e., some rows in $C$ may be zero. It happens when the corresponding node does not connect to any capacitor. Similarly, $L$ may be singular too, and it happens when the corresponding branch is a floating voltage source.

To go one step further to remove the extra variables in $s$ domain, (1.21) can be symbolically represented by

$$\begin{bmatrix} Cs & 0 \\ 0 & Ls \end{bmatrix} \begin{bmatrix} V \\ I \end{bmatrix} = - \begin{bmatrix} G_1 & W^T \\ -W & 0 \end{bmatrix} \begin{bmatrix} V \\ I \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix}, \quad (1.25)$$

or

$$\begin{bmatrix} G_1 + Cs & W^T \\ -W & Ls \end{bmatrix} \begin{bmatrix} V \\ I \end{bmatrix} = \begin{bmatrix} B \\ 0 \end{bmatrix}. \quad (1.26)$$

Using block Gauss elimination, $I$ can be first written as

$$I = (Ls)^{-1} W. \quad (1.27)$$

Replace $I$ in (1.26) with (1.27),

$$\left(\left[G_1 + Cs\right] + \left[W^T (Ls)^{-1} W\right]\right)\left[V\right] = \left[B\right] \quad (1.28)$$

## 1.4 Remarks

We reviewed three kinds of circuit analysis approaches: 1) $RC$ formulation, 2) $RLC$ formulation, and 3) general $RLC$ formulation. The first one, also known as nodal analysis, is widely used in circuit simulation tools such as SPICE[63] for its robustness and simplicity in implementation. The second formulation method is augmented based on the $RC$ formulation and it is different from the well-known modified nodal analysis, for it does not introduce current variables into the formulation. Therefore, our $RLC$ formulation is more compact, and has the nice property that it guarantees the non-singularity of matrix $M$ in (1.22) under certain assumptions. The last one is to be used on general $RLC$ circuits which may also contain mutual inductance. Further simplification can be done to reduce the matrix size in $s$ domain.

In the next subsection, we will give a closed form for circuit responses to $RC$ and $RLC$ formulations that we have introduced. In some ill cases, matrix $M$ in (1.22) may be singular. When it happens, the closed form solution will not be available. Therefore we reiterate our assumptions to the $RC$ and $RLC$ circuits of our interest: when an inductor is present in circuits, there has to be a series resistor with it; each inter-node has a coupling or ground capacitor. Under these assumptions, $M$ is non-singular.

## 2. Responses in Time Domain

### 2.1 Responses in Closed Form

Before proceeding to $s$ domain analysis, we discuss the time domain solution of linear networks derived from $RC$ and $RLC$ formulations in the previous subsection. Let us pre-multiply matrix $M^{-1}$ on both sides of (1.22); then we have

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= -M^{-1}G\boldsymbol{x}(t) + M^{-1}P\boldsymbol{u}(t) \\ &\equiv A\boldsymbol{x}(t) + B\boldsymbol{u}(t), \end{aligned} \tag{2.1}$$

where $A \equiv -M^{-1}G$ and $B \equiv M^{-1}P$. Pre-multiply $e^{-At}$ on both sides, (2.1) can be written as

$$\begin{aligned} e^{-At}\dot{\boldsymbol{x}}(t) - e^{-At}A\boldsymbol{x}(t) &= e^{-At}B\boldsymbol{u}(t) \\ \frac{\mathrm{d}\left[e^{-At}\boldsymbol{x}(t)\right]}{\mathrm{d}t} &= e^{-At}B\boldsymbol{u}(t) \end{aligned} \tag{2.2}$$

The solution to the above differential equation is the time-domain response on circuit nodes:

$$\boldsymbol{x}(t) = \boldsymbol{x}_0 + \int_{t_0}^{t} e^{A(t-\tau)}B\boldsymbol{u}(\tau)\mathrm{d}\tau \tag{2.3}$$

where $\boldsymbol{x}_0$ is the initial condition, i. e., $\boldsymbol{x}_0 = \boldsymbol{x}(t)|_{t=t_0}$.

The output response in time domain is derived from (2.3) by pre-multiplying matrix $Q$:

$$\boldsymbol{y}(t) = Q\boldsymbol{x}(t) = Q\boldsymbol{x}_0 + Q\int_{t_0}^{t} e^{A(t-\tau)}B\boldsymbol{u}(\tau)\,\mathrm{d}\tau \tag{2.4}$$

The first term is the output at time $t = t_0$. The second term is the convolution of the impulse response and the input waveform. The result can be verified by Laplace and inverse Laplace transformations.

### 2.2 Taylor Expansion in Time Domain

The matrix exponential $e^{At}$ is defined by Taylor expansion:

$$e^{At} = 1 + At + \frac{(At)^2}{2!} + \frac{(At)^3}{3!} + \cdots + \frac{(At)^n}{n!} + \cdots. \tag{2.5}$$



Therefore, the solution (2.3), with $t_0 = 0$ without losing any generality, can be rewritten as:

$$\begin{aligned} \boldsymbol{x}(t) &= \boldsymbol{x}_0 + \int_0^t e^{A(t-\tau)}B\boldsymbol{u}(\tau)\,\mathrm{d}\tau \\ &= \boldsymbol{x}_0 + \int_0^t \sum_{i=0}^{\infty} \frac{(A(t-\tau))^i}{i!}B\boldsymbol{u}(\tau)\,\mathrm{d}\tau \\ &= \boldsymbol{x}_0 + \sum_{i=0}^{\infty} \frac{A^iB}{i!}\int_0^t (t-\tau)^i\boldsymbol{u}(\tau)\,\mathrm{d}\tau. \end{aligned} \tag{2.6}$$

If we approximate $\boldsymbol{x}(t)$ by first $k$ terms:

$$\boldsymbol{x}(t) \cong \boldsymbol{x}_0 + \sum_{i=0}^{k-1} \frac{A^iB}{i!}\int_0^t (t-\tau)^i\boldsymbol{u}(\tau)\,\mathrm{d}\tau \tag{2.7}$$

Define

$$a_i \equiv \frac{A^iB}{i!} \tag{2.8}$$

$$\boldsymbol{x}_i(t) \equiv \int_0^t (t-\tau)^i\boldsymbol{u}(\tau)\,\mathrm{d}\tau \tag{2.9}$$

Therefore,

$$\boldsymbol{x}(t) = \boldsymbol{x}_0 + \sum_{i=0}^{k-1}\left[a_i\boldsymbol{x}_i(t)\right]. \tag{2.10}$$

We can get all the $a_i$ by $k$ Matrix-Vector multiplications. If the matrix is in Harwell-Boeing Format, the complexity of Matrix-Vector multiplication grows linearly with the number of non-zero elements in Matrix.

The evaluation of $\boldsymbol{x}_i(t)$ needs to take the source vector $U(t)$ into account. For a vector of constant sources, $\boldsymbol{u}(t) = \alpha$,

$$\boldsymbol{x}_i(t) = \frac{t^{i+1}}{i+1}\alpha. \tag{2.11}$$

For a vector of linear sources, $\boldsymbol{u}(t) = \alpha t$,

$$\boldsymbol{x}_i(t) = \frac{t^{i+2}}{(i+1)(i+2)}\alpha. \tag{2.12}$$

Sources with classical waveforms, such as exponential or sinusoidal function, have also closed form representation of (2.10). Some sources, on the other hand, are combinations of different ones mentioned above. One of its kind is piece-wise linear voltage or current source, which is a combination of a series of timed

ramp inputs. Because system analyzed here is a linear network, if system has different kinds of source, we can calculate $x_i(t)$ for each independent source alone and sum the response together.

Please note that $a_i$ and $b_i$ have no relation to source $u(t)$ and time $t$, thus $a_i$ and $b_i$ need to be computed only once. For each interested time point $t$, calculate the $x_i(t)$, substitute into (2.10), we can get the result value.

The choice of $k$ will greatly affect the accuracy of this method. For a given error tolerance, we want to find out the smallest $k$ that satisfies the accuracy requirement. Since the value of $u(t)$ is bounded in real circuit(e. g., less than 5V), We consider $u(t)$ is constant $\alpha$. Substitute (2.11) into (2.10), local truncation error $LTE$ can be approximated as:

$$\begin{aligned} LTE &\cong \sum_{i=k}^{\infty} \frac{(\| A \|)^i \| B\alpha \| t^{i+1}}{(i+1)!} \\ &< \frac{\| A \|^k \| B\alpha \| t^{k+1}}{(k+1)!(1-\frac{\|A\|t}{k+2})}. \end{aligned} \tag{2.13}$$

Here $\| \cdot \|$ is 1-norm($\| \cdot \|_1$) or $\infty$-norm($\| \cdot \|_\infty$) of matrix. $\frac{\|A\|t}{k+2}$ must be smaller than 1, otherwise the local truncation error does not converge. Because $\| A \|$ is a fixed value for a given circuit, $k$ and $t$ can be adjusted mutually to satisfy the convergence condition. Specifically, if $t$ is equal to $T$, the time point when circuit response is desired, then $k$ has to be large enough such that $k+2$ is greater than $\| A \| T$. On the other hand, if $k$ is set to a fixed value, e. g., 100, $T$ may have to be time stepped such that individual time step $t$ is small enough to make $\| A \| t$ smaller than $k+2$. In summary, the smaller $t$ is, the smaller $k$ could be.

For a given time point $T$, the absolute truncation error $ATE = \frac{T}{t} LTE$, i. e.,

$$ATE < \frac{\| A \|^k \| B\alpha \| t^k}{(k+1)!(1-\frac{\|A\|t}{k+2})}. \tag{2.14}$$

Theoretically because self-multiplication of matrix $A$ is more expensive than matrix-vector multiplication in (2.10), and $A^i B$ is much smaller than the square matrix $A$ in terms of dimension, we can select time step $t$ as small as possible. Thus for the same absolute truncation error, smaller $k$ is allowed.

For non-stiff systems, Taylor expansion method can approach the accuracy of SPICE with about one or two order less computing time. For stiff systems, this method requires small time steps compared to the interested time interval. It will generate extremely long simulation time. Practically, $k = 10$ is a reasonable number for most of systems.



## 3. $s$ Domain Analysis

In this section, we discuss how to obtain the transfer function matrix of a linear network from the modified nodal analysis formulation in $s$ domain, and how to convert the responses in $s$ domain to the time domain.

### 3.1 Transfer Function

The Laplace transformation of the modified nodal analysis equations (1.22) and (1.23) is given by

$$sM\mathbf{x}(s) - M\mathbf{x}_0 = -G\mathbf{x}(s) + Pu(s) \tag{3.1}$$

$$\mathbf{y}(s) = Q\mathbf{x}(s). \tag{3.2}$$

Recall that $\mathbf{x}_0$ is the initial condition of the time domain vector $\mathbf{x}(t)$. Premultiply $G^{-1}$ on both sides of (3.1) and obtain

$$(I + sG^{-1}M)\mathbf{x}(s) = G^{-1}Pu(s) + G^{-1}M\mathbf{x}_0, \tag{3.3}$$

or

$$(I - sA)\mathbf{x}(s) = Bu(s) + C\mathbf{x}_0, \tag{3.4}$$

where

$$A \equiv -G^{-1}M \quad B \equiv G^{-1}P \quad C \equiv G^{-1}M\mathbf{x}_0. \tag{3.5}$$

Therefore, we can derive the solution to (3.1) as

$$\mathbf{x}(s) = (I - sA)^{-1}Bu(s) + (I - sA)^{-1}C\mathbf{x}_0. \tag{3.6}$$

Insert (3.6) into (3.2); the output, $\mathbf{y}(s)$, can be represented by

$$\mathbf{y}(s) = Q(I - sA)^{-1}Bu(s) + Q(I - sA)^{-1}C\mathbf{x}_0. \tag{3.7}$$

So the transfer function matrix defining the relationship between the input $u(s)$ and the output $\mathbf{y}(s)$ is given by

$$H(s) = Q(I - sA)^{-1}B. \tag{3.8}$$

**Example 2.4.** To interpret the definition of the transfer function matrix $H(s)$, let us consider a linear network with two input terminals and three output terminals shown in Fig. 2.5. The I/O terminals are related by the transfer function matrix below:

$$\begin{bmatrix} Y_1(s) \\ Y_2(s) \\ Y_3(s) \end{bmatrix} = \begin{bmatrix} H_{11}(s) & H_{12}(s) \\ H_{21}(s) & H_{22}(s) \\ H_{31}(s) & H_{32}(s) \end{bmatrix} \begin{bmatrix} U_1(s) \\ U_2(s) \end{bmatrix}. \tag{3.9}$$

In the transfer function matrix, $H_{ij}(s)$ is the impulse response at output $Y_i$ when $U_j$ has an impulse input and the other input terminal is off[4].

[4]To turn off the input terminal, if it is connected to a voltage source, it has to be grounded; if connected to a current source, it has to be disconnected.

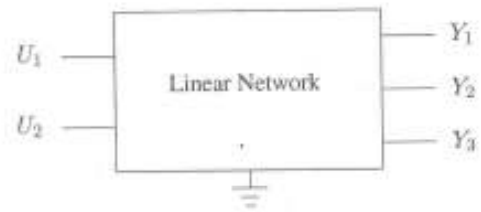



*Figure 2.5.* A linear network with two inputs and three outputs.

## 3.2 Responses from $s$ Domain to Time Domain

In general, matrix $H(s)$ in (3.8) would be a $M \times N$ matrix, where $M$ and $N$ are the numbers of inputs and outputs of the system, respectively. Due to term $(I - sA)^{-1}$ in (3.8), each entry in $H(s)$ is a real rational function of $s$, i. e.,

$$H_{ij}(s) = \frac{Y_i(s)}{U_j(s)} \tag{3.10}$$

$$= \frac{a_0 + a_1 s + a_2 s^2 + \cdots + a_m s^m}{b_0 + b_1 s + b_2 s^2 + \cdots + b_n s^n} \tag{3.11}$$

$$= \frac{a_m (s - z_1)(s - z_2) \cdots (s - z_m)}{b_n (s - p_1)(s - p_2) \cdots (s - p_n)}, \tag{3.12}$$

where $a_i$ and $b_i$ are real coefficients of the polynomial expressions of $s$, and $z_i$ and $p_i$ are the *zeros* and *poles* of the transfer function, respectively.

Furthermore, all the entries share the same denominator. In fact, the partial fraction decomposition of $H_{ij}(s)$ is given by

$$H_{ij}(s) = \frac{k_1}{s - p_1} + \frac{k_2}{s - p_2} + \cdots + \frac{k_n}{s - p_n}, \tag{3.13}$$

or

$$H_{ij}(s) = \frac{r_1}{1 + s\lambda_1} + \frac{r_2}{1 + s\lambda_2} + \cdots + \frac{r_n}{1 + s\lambda_n}, \tag{3.14}$$

where $\lambda_i$ is the $i$-th eigenvalue of square matrix $A_{n \times n}$. It is worth noting that $p_i = -1/\lambda_i$, which is the relationship between system poles and eigenvalues.

Particularly, from the expression in partial fraction decomposition, the time domain impulse response at output terminal $Y_j$ to the impulse input at input terminal $U_i$ can be obtained via inverse Laplace transformation:

$$y_j(t) = k_1 e^{p_1 t} + k_2 e^{p_2 t} + \cdots + k_n e^{p_n t}. \tag{3.15}$$

For an arbitrary input signal at $U_i$, performing convolution on the impulse response and the signal gives us the time domain response at $Y_j$. For arbitrary input signals at all input terminals, time domain output responses at $Y_j$ can be obtained via principle of superposition.



# 4. Preliminaries of Symbolic Analysis

In this section, we briefly review some mathematic notations and theories relevant to the graph-based symbolic analysis techniques to be discussed in details in the later chapters.

## 4.1 Matrix, Determinant, and Cofactors

Let $e = \{1, ..., n\}$ be a set of integers. Let $A$ denote a set of $m$ elements, called *symbolic parameters* or simply *symbols*, $\{a_1, ..., a_m\}$, where $1 \leq m \leq n^2$ and each symbol is labeled by a unique pair $(r, c)$, where $r \in e$ and $c \in e$. Often, we write $A$ as an $n \times n$ (square) *matrix*, denoted by $\mathbf{A}$, and use $a_{r,c}$ to denote the element of matrix $\mathbf{A}$ at row $r$ and column $c$. We sometimes use $r(a)$ and $c(a)$ to denote, respectively, the row and column indices of element $a$.

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{bmatrix}.$$

If $m = n^2$ the matrix is said to be *full*. If $m << n^2$ the matrix is said to be *sparse*. The *determinant* of $\mathbf{A}$, denoted by $det(\mathbf{A})$, is defined by

$$\begin{vmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{vmatrix} = \sum_{j_1 \neq j_2 \neq \ldots \neq j_n} (-1)^p \cdot a_{1,j_1} \cdot a_{2,j_2} \cdots a_{n,j_n}. \tag{4.1}$$

Here $(j_1, j_2, ..., j_n)$ is a permutation of $e$, and $p$ is the number of permutations needed to make the sequence $(j_1, j_2, ..., j_n)$ monotonically increasing. The right hand side of (4.1) is a symbolic expression of $det(\mathbf{A})$ in the *expanded* form, more precisely, the *sum-of-product* form, where each *term* is an algebraic product of $n$ symbolic parameters. We note that each symbol can be assigned a real or complex value for analog circuit simulation.

Let $p$, $p \subseteq e$, and $q$, $q \subseteq e$ such that $|p| = |q|$. The square matrix obtained from the matrix $\mathbf{A}$ by deleting those rows not in $p$ and columns not in $q$ forms a *submatrix* of $\mathbf{A}$, and is represented by $\mathbf{A}(p, q)$. It has dimension $|p|$ by $|q|$.

Let $a_{r,c}$ be the element of $\mathbf{A}$ at row $r$ and column $c$. Let $\mathbf{A}_{a_{r,c}}$ be the $(n-1) \times (n-1)$-matrix obtained from the matrix $\mathbf{A}$ by deleting row $r$ and column $c$, and let $\mathbf{A}_{\bar{a}_{r,c}}$ be the $n \times n$-matrix obtained from $\mathbf{A}$ by setting $a_{r,c} = 0$. Then, the determinant of matrix $\mathbf{A}$ can be *expanded* as below in a way similar to Shannon expansion for Boolean functions:

$$det(\mathbf{A}) = a_{r,c}(-1)^{r+c} det(\mathbf{A}_{a_{r,c}}) + det(\mathbf{A}_{\bar{a}_{r,c}}), \tag{4.2}$$

where $(-1)^{r+c} det(\mathbf{A}_{a_{r,c}})$ is referred to as the *cofactor* of $det(\mathbf{A})$ with respect to $a_{r,c}$, and $det(\mathbf{A}_{\bar{a}_{r,c}})$ as the *remainder* of $det(\mathbf{A})$ with respect to $a_{r,c}$. The

Chapter 3

# MODEL-ORDER REDUCTION

## 1. $s$ Domain Analysis

In this section, we discuss how to obtain the transfer function matrix of a linear network from the MNA formulation in $s$ domain, and how to convert the responses in $s$ domain to the time domain.

## 2. Moments and Moment-Matching Method

### 2.1 Concept of Moments

In $s$ domain, since the Laplace transform of the impulse function, $\delta(t)$, is unity[1], the response at a port is the transfer function itself.

**Definition 3.1 (Moments of Impulse Response).** The moments of an impulse response are the coefficients of powers of $s$ in Maclaurin expansion of the transfer function, $H(s)$, in $s$ domain, i. e.,

$$H(s) = \sum_{k=0}^{\infty} m_k s^k, \tag{2.1}$$

[1] Impulse function is defined as:

$$\delta(t) = \begin{cases} \infty & t = 0 \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad \int_{0^-}^{0^+} \delta(t)\mathrm{d}t \equiv 1.$$

Therefore, its Laplace transform is given by

$$\int_{0^-}^{\infty} \delta(t)e^{-st}\mathrm{d}t = \int_{0^-}^{0^+} \delta(t)\mathrm{d}t = 1$$

where

$$m_k = \frac{1}{k!} \times \left. \frac{\mathrm{d}^k H(s)}{\mathrm{d}s^k} \right|_{s=0}. \tag{2.2}$$

## 2.2 Delay Estimation Using Moments

moments, delay In this section, we examine the connections of moments and delay estimation of interconnect circuits in two-fold. First, we investigate the reason why moments give good measurement for delay estimation. Secondly, we see how to approximate delays, provided that a set of moments are already given.

### 2.2.1 Moments: Characteristics of Impulse Response

Moments, metric in $s$ domain, are tightly related to the impulse response waveform in time domain. Indeed, they characterize the shape of the waveform. Let $H(t)$ be the impulse response in the time domain, we rewrite moments defined in (2.2) in terms of $H(t)$ by using Maclaurin expansion of $e^{-st}$ in the Laplace transform $H(s)$; i. e.,

$$\begin{aligned} H(s) &= \int_0^\infty H(t)e^{-st}\mathrm{d}t \\ &= \int_0^\infty \left(1 - st + s^2\frac{t^2}{2} + \cdots + s^k\frac{(-1)^k}{k!}t^k + \cdots\right)\mathrm{d}t \\ &= \sum_{k=0}^{\infty} s^k \frac{(-1)^k}{k!}\int_0^\infty t^k H(t)\mathrm{d}t \end{aligned} \tag{2.3}$$

Comparing (2.3) with the definition that $H(s) = \sum_{k=0}^{\infty} m_k s^k$, moments can be rewritten as:

$$m_k = \frac{(-1)^k}{k!}\int_0^\infty t_k H(t)\mathrm{d}t, \tag{2.4}$$

or

$$m_0 = \int_0^\infty H(t)\mathrm{d}t \tag{2.5}$$

$$m_1 = -\int_0^\infty tH(t)\mathrm{d}t \tag{2.6}$$

$$\cdots$$

$m_0$ is the total area under the response curve $H(t)$, which is unity in the case that no resistive load is grounded. We will discuss more on this in Section 2.4. load.

Since the requirement is applicable to most of the linear circuits in the thesis, we can assume that $\int_0^\infty H(t)\mathrm{d}t = 1$. The mean of the impulse response,



$\int_0^\infty tH(t)\mathrm{d}t / \int_0^\infty H(t)\mathrm{d}t = -m_1$, is widely used as an approximation of step response delays. As the step response in $s$ domain is given by $\frac{1}{s}H(s)$, basic Laplace transformation properties tell us that the step response in the time domain is $\int_0^\infty H(t)\mathrm{d}t$, as shown in Fig. 3.1. Therefore, the 50% delay of $Y(t)$

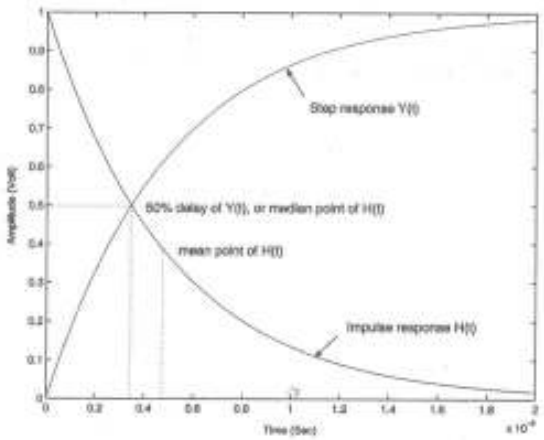



*Figure 3.1.* Scenario that the median and mean points mismatch: when the unit impulse response $H(t)$ (scaled) is not symmetric, 50% delay of unit step response $Y(t)$, or the median point of $H(t)$, does not overlap with the mean point of $H(t)$.

is essentially the median point of the unit impulse response. Furthermore, if $H(t)$ is symmetric, the mean of the impulse response is exactly the median point, i. e., 50% delay of $Y(t)$ is accurately $-m_1$ (Fig. 3.2). The first moment of the impulse response also known as Elmore delay, is used as a dominant time constant approximation for RC trees. Indeed, $m_1$ provides an upper bound of delays for *RC* trees due to the resistance shielding effect[67].

Higher order moments give more sophisticated measurement for the distribution of the impulse response [52]. Therefore, functions that match moments of the impulse response are expected to give a good approximation to the waveform.

### 2.2.2 Padé Approximation

Provided that a set of moments of the impulse response are given, Padé approximation is a method that generates a family of rational functions whose moments agree with those of the impulse response. The rational functions are further decomposed into partial fractions, whose inverse Laplace transforms are used to constitute the approximated response waveforms.

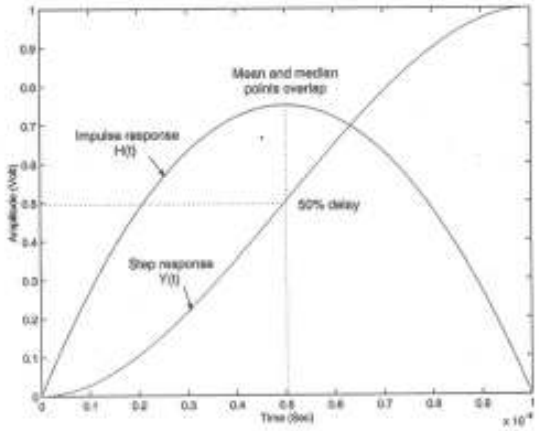



*Figure 3.2.* Scenario that the median and mean points of $H(t)$ overlap: when $H(t)$ is symmetric, 50% delay point of $Y(t)$ matches the mean point of $H(t)$, i. e., the approximation is exact.

**Definition 3.2 (Padé Approximation).** Given two integers $p$ and $q$, $(p, q)$ Padé approximation of the transfer function $H(s)$ is a rational function

$$H_{p,q}(s) = \frac{P(s)}{Q(s)} = \frac{a_0 + a_1 s + a_2 s^2 + \cdots + a_p s^p}{1 + b_1 s + b_2 s^2 + \cdots + b_q s^q}. \tag{2.7}$$

The Maclaurin expansion of $H_{p,q}(s)$ agrees with the that of $H(s)$ in the first $p + q + 1$ terms, i. e.,

$$H(s) = H_{p,q}(s) + O(s^{p+q+1}). \tag{2.8}$$

As there are $p + q + 1$ unknowns in (2.7), we need to establish $p + q + 1$ independent equations to solve for them. Assuming that $H_{p,q}(s)$ is a proper transfer function, i. e., $p < q$, we can get coefficients in denominator $Q(s)$ of (2.7) by solving the following equations:

$$\begin{bmatrix} 0 & \cdots & 0 & 0 & m_0 & m_1 & \cdots & m_p \\ 0 & \cdots & 0 & m_0 & m_1 & m_2 & \cdots & m_{p+1} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ m_0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & m_{q-1} \\ m_1 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & m_q \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ m_p & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & m_{p+q-1} \end{bmatrix} \begin{bmatrix} b_q \\ b_{q-1} \\ \vdots \\ b_1 \end{bmatrix} = - \begin{bmatrix} m_{p+1} \\ m_{p+2} \\ \vdots \\ m_q \\ m_{q+1} \\ \vdots \\ m_{p+q} \end{bmatrix}. \tag{2.9}$$



And the coefficients $a_k$ of numerator $Q(s)$ satisfy the equation:

$$\begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} m_0 & 0 & 0 & \cdots & 0 \\ m_1 & m_0 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ m_p & m_{p-1} & m_{p-2} & \cdots & m_0 \end{bmatrix} \begin{bmatrix} 1 \\ b_1 \\ b_1 \\ \vdots \\ b_p \end{bmatrix}. \tag{2.10}$$

Equations (2.9) and (2.10) can be verified by honoring the fact that the first $p + q + 1$ moments of $H_{p,q}(s)$ match those of $H(s)$, i. e.,

$$\begin{aligned} \frac{P(s)}{Q(s)} &= \frac{a_0 + a_1 s + a_2 s^2 + \cdots + a_p s^p}{1 + b_1 s + b_2 s^2 + \cdots + b_q s^q} \\ &= m_0 + m_1 s + \cdots + m_{p+q} s^{p+q} + r(s) s^{p+q+1}, \end{aligned} \tag{2.11}$$

where $r(s)$ is a polynomial function of $s$. Multiplying both sides with denominator $Q(s)$, we have

$$\begin{aligned} a_0 + a_1 s + a_2 s^2 + \cdots + a_p s^p = &\left(1 + b_1 s + b_2 s^2 + \cdots + b_q s^q\right) \\ &\times \left(m_0 + m_1 s + \cdots + m_{p+q} s^{p+q} + r(s) s^{p+q+1}\right). \end{aligned} \tag{2.12}$$

By equating the coefficients of powers of $s$ on both sides, we are able to write the two equations in (2.9) and (2.10).

**Example 3.1.** Given a set of moments

$$m_0 = 1, \quad m_1 = 2, \quad m_2 = 5, \quad m_3 = 20, \tag{2.13}$$

to obtain a (1,2) Padé approximation, (2.9) can be instantiated as

$$\begin{bmatrix} m_0 & m_1 \\ m_1 & m_2 \end{bmatrix} \begin{bmatrix} b_2 \\ b_1 \end{bmatrix} = - \begin{bmatrix} m_2 \\ m_3 \end{bmatrix}.$$

Inserting (2.13) to the equation above, we have

$$\begin{bmatrix} 1 & 2 \\ 2 & 5 \end{bmatrix} \begin{bmatrix} b_2 \\ b_1 \end{bmatrix} = - \begin{bmatrix} 5 \\ 20 \end{bmatrix}.$$

Therefore, $b_1 = -10$ and $b_2 = 15$. And to get the numerator, (2.10) is instantiated as

$$\begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = \begin{bmatrix} m_0 & 0 \\ m_1 & m_0 \end{bmatrix} \begin{bmatrix} 1 \\ b_1 \end{bmatrix},$$

or

$$\begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ -10 \end{bmatrix}.$$

So, $a_0 = 1$ and $a_1 = -8$. Thus,

$$H_{1,2}(s) = \frac{P(s)}{Q(s)} = \frac{1 - 8s}{1 - 10s + 15s^2}. \tag{2.14}$$

### 2.2.3 Partial Fraction Decomposition

In order to get the full response waveform, after getting rational function $H_{p,q}(s)$ from Padé approximation, one needs to derive the partial fraction decomposition of it. Assuming that the input is $V_{in} = 1/(s - p_0)$, the partial fraction decomposition of $V_{out}$ is given by

$$\begin{aligned} V_{out} &= H_{p,q}V_{in} = \frac{P(s)}{Q(s)}\frac{1}{s-p_0} \\ &= \frac{k_0}{s-p_0} + \frac{k_1}{s-p_1} + \cdots + \frac{k_p}{s-p_q} \\ &= \sum_{i=0}^{q} \frac{k_i}{s-p_i}. \end{aligned} \tag{2.15}$$

The $q$ roots of $Q(s)$, $p_1, \cdots, p_q$, are obtained by solving the polynomial function $Q(s)$ directly. And for the residues $k_j$ in (2.15), since

$$\frac{P(s)}{\prod_{i=0}^{q}(s-p_i)} = \sum_{i=0}^{q} \frac{k_i}{s-p_i},$$

we multiply factor $(s - p_j)$ on both sides and obtain

$$\frac{P(s)}{\prod_{i=0,i\neq j}^{q}(s-p_i)} = \sum_{i=0,i\neq j}^{q} \frac{k_i}{s-p_i}(s-p_j) + k_j.$$

To derive $k_j$, we substitute $s$ for $p_j$ in the above equation, thus

$$k_j = \frac{P(p_j)}{\prod_{i=0,i\neq j}^{q}(p_j-p_i)} \tag{2.16}$$

**Example 3.2.** Continuing Example 3.1, we evaluate the partial fraction decomposition of $H_{1,2}(s)$ in (2.14). The two roots of $Q(s)$ are $p_1 = 0.12$ and $p_2 = 0.54$. Assuming that the input is $V_{in} = 1/(s - 2)$, then

$$\begin{aligned} k_0 &= \left.\frac{1-8s}{(s-0.12)(s-0.54)}\right|_{s=2} = -0.37 \\ k_1 &= \left.\frac{1-8s}{(s-0.54)(s-2)}\right|_{s=0.12} = -0.0017 \\ k_2 &= \left.\frac{1-8s}{(s-0.12)(s-2)}\right|_{s=0.54} = 0.36. \end{aligned}$$

Thus,

$$V_{out} = H_{1,2}V_{in} = -\frac{0.37}{s-2} - \frac{0.0017}{s-0.12} + \frac{0.36}{s-0.54}.$$



## 2.3 Deriving Moments from MNA Formulation

Definition 2.2 can be readily extended to the matrix form for multi-port systems (e. g., Fig. 2.5). We do so by examining the MNA formulation method given in Section 1.2. We rewrite the formulation in time domain (1.22) and (1.23) here:

$$\begin{aligned} M\dot{\boldsymbol{x}}(t) &= -G\boldsymbol{x}(t) + P\boldsymbol{u}(t) \\ \boldsymbol{y}(t) &= Q\boldsymbol{x}(t). \end{aligned}$$

Assuming that $\boldsymbol{x}(0) = 0$, the Laplace transformation of the above two equations are given by

$$\begin{aligned} sM\boldsymbol{x}(s) &= -G\boldsymbol{x}(s) + P\boldsymbol{u}(s) \\ \boldsymbol{y}(s) &= Q\boldsymbol{x}(s), \end{aligned}$$

or

$$\boldsymbol{x}(s) = (G + sM)^{-1}P\boldsymbol{u}(s) \tag{2.17}$$

$$\boldsymbol{y}(s) = Q\boldsymbol{x}(s). \tag{2.18}$$

By substituting $\boldsymbol{x}(s)$ for (2.17), we can write $\boldsymbol{y}(s)$ as:

$$\boldsymbol{y}(s) = Q(G + sM)^{-1}P\boldsymbol{u}(s). \tag{2.19}$$

The transfer function in (2.19) is defined by

$$H(s) \equiv Q(G + sM)^{-1}P. \tag{2.20}$$

And refer to Definition 2.2, moments of $H(s)$ in (2.20), i. e., the coefficients of Maclaurin expansion of $H(s)$ are given by:

$$M_j = (-1)^j Q(G^{-1}M)^j G^{-1}P, \tag{2.21}$$

where $0 \leq j \leq \infty$.

Computation of moments requires $G$ to be invertible. This requirement is easily satisfied by most interconnect circuits in which each node has a DC path to the ground.

## 2.4 Deriving Moments for RLC Trees

In the previous section we showed the general approach to computing moments of any linear circuits. In this section, we demonstrate the ease of moment computations for a family of special linear circuits, RLC trees.

In (2.21), suppose we consider all the entries in the unknown vector $\mathbf{x}(s)$ as outputs, i. e., $Q$ is an identity matrix, we have

$$M_0 = G^{-1}PU \tag{2.22}$$
$$M_1 = G^{-1}MG^{-1}P = G^{-1}MM_0 \tag{2.23}$$
$$M_2 = (G^{-1}M)^2G^{-1}P = G^{-1}MM_1 \tag{2.24}$$
$$\cdots$$

We find out from the equations above, that $M_0$ can be solved in a linear equation

$$GM_0 = PU. \tag{2.25}$$

And furthermore, higher order moments can be evaluated by utilizing the previous ones, i. e.,

$$GM_1 = MM_0 \tag{2.26}$$
$$GM_2 = MM_1 \tag{2.27}$$
$$\cdots$$
$$GM_{i+1} = MM_i \tag{2.28}$$
$$\cdots$$

We investigate on how to evaluate moments iteratively. First of all, let us start with $M_0$. To solve for $M_0$ in (2.25, we notice that matrix $G$ is the admittance matrix of a resistive tree derived from the original RLC tree by removing all the capacitors and inductors. The inputs, however, are kept unchanged. Fortunately, for most of the RLC trees or tree-like circuits in the thesis-wise scope, the DC solution is trivial. We will show this in an example later.

$M_i$ now is supposedly given, let look at how to derive $M_{i+1}$ from $M_i$. Refer to (2.28), matrix $G$ is not changed, i. e., again we need to solve the DC solution of the resistive tree. However, the system's inputs are now changed to $MM_i$. In (1.24) we showed that $M$ is in the form of

$$M \equiv \begin{bmatrix} C & 0 \\ 0 & L \end{bmatrix},$$

where matrix $C$ is the conductive matrix, and $L$ is the inductive matrix. Due to the way that MNA formulation defines the two matrices, entries in $C$ correspond to the currents flowing through capacitors, and entries in $L$ correspond to the voltages across inductors. In other words, $C$ is a part of the KCL formulation, while $L$ is a part of the KVL formulation. Therefore, if we partition $MM_i$ according to the composition of $M$,

$$MM_i = \begin{bmatrix} C & 0 \\ 0 & L \end{bmatrix} \begin{bmatrix} M_{iV} \\ M_{iI} \end{bmatrix} = \begin{bmatrix} I_C \\ V_L \end{bmatrix}, \tag{2.29}$$



then in $MM_i$, which is also the right-hand side (RHS) of (2.28), $I_C$ is a vector of current sources, and $V_L$ is a set of voltage sources. An entry in $I_C$ is a product of capacitance and the $i$-th moment of the voltage ($M_{iV}$) across it, and an entry in $V_L$ is a product of inductance and the $i$-th moment of the current ($M_{iI}$) through it. Accordingly, we can generate a new "resistive tree" from the old one by adding current sources and voltage sources at locations of capacitors and inductors of the original RLC tree, respectively, and zeroing out the voltage sources in the old "resistive tree". The solution to such circuit is also trivial: one can evaluate branch currents and voltage drops in an inverse breadth-first-search (BFS) fashion, starting from the leaf nodes; when the root is reached, a BFS or DFS (depth-first-search) can be performed from the root to the leaves to update node voltages.

**Example 3.3.** The circuit given in Fig. 3.3 is a general RLC tree. The input $V_{in}$ is an unit impulse function. An important property of the RLC tree is that each node in the tree has a DC path to the ground, and this path has to go through the voltage source. In other words, no resistors or inductors are connected to the ground directly.

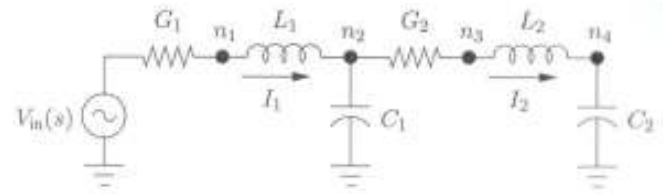



*Figure 3.3.* The original RLC tree in Example 3.3.

The symbolic MNA formulation of the circuit is given by

$$\begin{bmatrix} G_1 & 0 & 0 & 0 & 1 & 0 \\ 0 & G_2+C_1s & -G_2 & 0 & -1 & 0 \\ 0 & -G_2 & G_2 & 0 & 0 & 1 \\ 0 & 0 & 0 & C_2s & 0 & -1 \\ -1 & 1 & 0 & 0 & L_1s & 0 \\ 0 & 0 & -1 & 1 & 0 & L_2s \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \\ I_1 \\ I_2 \end{bmatrix} = \begin{bmatrix} V_{in}G_1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{2.30}$$

The above equation is written in $s$ domain. Comparing it with the time domain counterpart in (1.22), we have

$$M = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & C_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & C_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & L_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & L_2 \end{bmatrix}, \quad G = \begin{bmatrix} G_1 & 0 & 0 & 0 & 1 & 0 \\ 0 & G_2 & -G_2 & 0 & -1 & 0 \\ 0 & -G_2 & G_2 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & -1 \\ -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 \end{bmatrix} \tag{2.31}$$

Each unknown in (2.30) is a rational function of $s$ due to Cramer's rule (more on this in the next chapter). And the Maclaurin expansion is given by

$$V_1 = m_0^{(V_1)} + m_1^{(V_1)} s + m_2^{(V_1)} s^2 + \cdots \tag{2.32}$$
$$V_2 = m_0^{(V_2)} + m_1^{(V_2)} s + m_2^{(V_2)} s^2 + \cdots \tag{2.33}$$
$$V_3 = m_0^{(V_3)} + m_1^{(V_3)} s + m_2^{(V_3)} s^2 + \cdots \tag{2.34}$$
$$V_4 = m_0^{(V_4)} + m_1^{(V_4)} s + m_2^{(V_4)} s^2 + \cdots \tag{2.35}$$
$$I_1 = m_0^{(I_1)} + m_1^{(I_1)} s + m_2^{(I_1)} s^2 + \cdots \tag{2.36}$$
$$I_2 = m_0^{(I_2)} + m_1^{(I_2)} s + m_2^{(I_2)} s^2 + \cdots \tag{2.37}$$

According to (2.25),

$$M_0 \equiv \begin{bmatrix} m_0^{(V_1)} & m_0^{(V_2)} & m_0^{(V_3)} & m_0^{(V_4)} & m_0^{(I_1)} & m_0^{(I_2)} \end{bmatrix}^T$$

is the solution to the equation

$$\begin{bmatrix} G_1 & 0 & 0 & 0 & 1 & 0 \\ 0 & G_2 & -G_2 & 0 & -1 & 0 \\ 0 & -G_2 & G_2 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & -1 \\ -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} m_0^{(V_1)} \\ m_0^{(V_2)} \\ m_0^{(V_3)} \\ m_0^{(V_4)} \\ m_0^{(I_1)} \\ m_0^{(I_2)} \end{bmatrix} = \begin{bmatrix} V_{\text{in}} G_1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{2.38}$$

Or if we go by inspection of the "resistive tree" (Fig. 3.4) obtained from Fig. 3.3 by removing all the capacitors and inductors. It is trivially seen that voltage at any node of the circuit is $V_{\text{in}}$, and there is no current from $n_1$ to $n_2$ or from $n_3$ to $n_4$. Therefore,

$$M_0 = \begin{bmatrix} V_{\text{in}} & V_{\text{in}} & V_{\text{in}} & V_{\text{in}} & 0 & 0 \end{bmatrix}^T. \tag{2.39}$$

We can verify that it *is* the solution to (2.38).



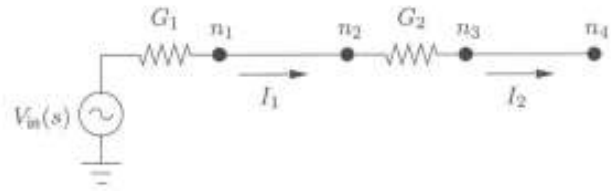



*Figure 3.4.* The "resistive tree" for computing $M_0$ in Example 3.3.

To get

$$M_1 \equiv \begin{bmatrix} m_1^{(V_1)} & m_1^{(V_2)} & m_1^{(V_3)} & m_1^{(V_4)} & m_1^{(I_1)} & m_1^{(I_2)} \end{bmatrix}^T,$$

we utilize (2.29):

$$\begin{bmatrix} G_1 & 0 & 0 & 0 & 1 & 0 \\ 0 & G_2 & -G_2 & 0 & -1 & 0 \\ 0 & -G_2 & G_2 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & -1 \\ -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} m_1^{(V_1)} \\ m_1^{(V_2)} \\ m_1^{(V_3)} \\ m_1^{(V_4)} \\ m_1^{(I_1)} \\ m_1^{(I_2)} \end{bmatrix} =$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & C_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & C_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & L_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & L_2 \end{bmatrix} \begin{bmatrix} m_0^{(V_1)} \\ m_0^{(V_2)} \\ m_0^{(V_3)} \\ m_0^{(V_4)} \\ m_0^{(I_1)} \\ m_0^{(I_2)} \end{bmatrix} = \begin{bmatrix} 0 \\ C_1 m_0^{(V_2)} \\ 0 \\ C_2 m_0^{(V_4)} \\ L_1 m_0^{(I_1)} \\ L_2 m_0^{(I_2)} \end{bmatrix}.$$

Accordingly, we modify Fig. 3.3 by zeroing out the voltage source $v_{\text{in}}$) and replacing capacitors with current sources and inductors with voltage sources. The modified circuit is depicted in Fig. 3.5. Again if we go by inspection, the solution to the new "resistive tree" is trivial: $I_2$ is equal to $-C_2 m_0^{(V_4)}$ and $V_{2,3}$ is uniquely determined by $I_2/G_2$; have gleaning all the down-stream currents of $n_2$, $I_1$ is simply $I_2 - C_1 m_0^{(V_2)}$... After we get all the branch currents and voltages, we need another tree-walk from the root to finally get *node* voltages, e. g., $V_1 = -I_1/G_1$, $V_2 = V_1 + V_{1,2}$, $V_3 = V_2 + V_{2,3}$, and $V_4 = V_3 + V_{3,4}$. Thus, these new node voltages and voltage-source currents constitute the first-order moment vector $M_1$. And these values are then used to evaluate new voltage and current source values for the next moment computation.

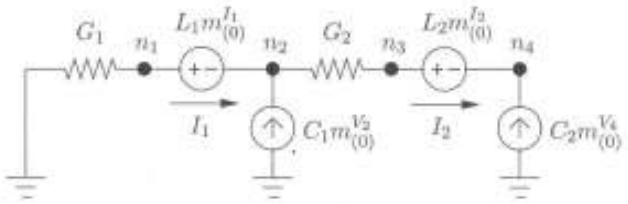



*Figure 3.5.* The "resistive tree" modified based on the original RLC circuit by zeroing out inputs and replacing capacitors and inductors with current and voltage sources, respectively.

### 2.4.1 Remarks

The moment-matching method is summarized as follows: given an RLC tree, one first computes a number of moments iteratively, each iteration is to get the DC solution to a "simplified" resistive tree-structured circuit. After the computation of moments, Padé approximation is used to find a rational function which matches the moments just evaluated. Finally, one can use partial fraction decomposition on the rational function and inverse Laplace transformation to get the approximated waveform. This work is the so-called asymptotic waveform evaluation method, or AWE, invented by Pillage and Rohrer [66] in 1990. The complexity of AWE method is $O(c \cdot n)$, i. e., it is linear in terms of both the number of moments desired ($c$) and the number of nodes in the circuit.

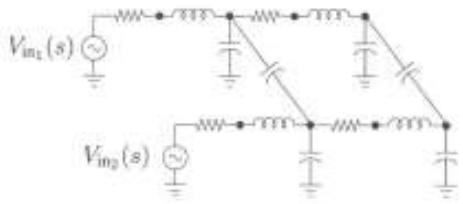
An explicit solution to the circuit with capacitors replaced by current sources and inductors by voltage sources —the so-called "companion network" —is also possible for circuit configuration other than strict RLC trees. Any such companion circuit for which a tree can be specified by only voltage sources or a co-tree can be specified by only current sources and no more current sources in the tree[2] has a trivial DC solution. For instance, the coupling interconnect circuit shown in Fig. 3.6 can be solved explicitly, because all the current sources in (b) are in the co-tree, and only resistors and voltage sources are in the tree.

## 3. Realizable Topological Reduction Methods

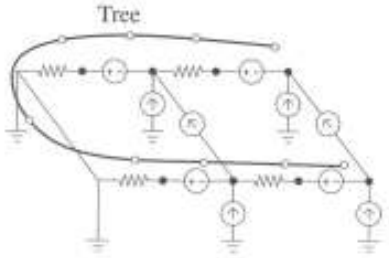
In the last section, we showed that moments provide good approximations to interconnect circuit responses. Essentially, the more moments are matched, the more accurate the approximation may become. And this is how AWE[66] is named. Although no rigorous proof could be given, it has already become a

[2]For a network for which a tree can be specified by only voltage sources, Then the node voltage can be trivially evaluated by a one-way tree-walk. This is due to KVL. On the other hand, for a network for which a co-tree can be specified by only current sources, then currents in any tree branch can be trivially solved. This is due to KCL. Furthermore, if no more current sources are in the tree, then voltage drops across tree branches, or equivalently node voltages, can be explicitly evaluated as well.






(a)




(b)

*Figure 3.6.* Illustration showing a type of non-tree circuit configuration having trivial DC solutions: (a) A RLC network; (b) the companion network of (a) for which all the links can be specified by current sources (or capacitors in the original network).

thumb rule. An obvious drawback of AWE, however, is that Padé approximation it uses may generate unstable poles[3].

To preserve the stability and passivity of the original circuit, realizable reduction is preferred.

**Definition 3.3 (Realizable Reduction).** A *realizable reduction* method preserves the stability and passivity of a given linear network by guaranteeing that the reduced models are realizable, i. e., all the RCLK elements in the reduced network are positive.

Realizability of impedances or admittances is a very hard problem in network synthesis. Realizability checking calls for a very complicated procedure called

[3]An unstable transfer function (system) has some pole(s) located on the right-half of the complex plane. Such poles are so-called unstable poles. For a stable system, if the input is bounded, the output must also be bounded. And this is not true for unstable systems. One can use partial fraction decomposition introduced in Section 2.2.3 to understand why systems with poles on the right-half complex plane are unstable.

"positive real". We will postpone the definition to Chapter 7. In this section, we review some realizable reduction techniques. Realizable reduction have been attracting a lot of attention from researchers. Although these techniques are state-of-the-art, they still impose various limitations on configurations of reducible circuits.

## 3.1 TICER

Sheehan proposed TICER —a realizable reduction method for RC circuits in 1999[74]. In Sheehan's implementation, a $n$-terminal star network is considered. A branch consists of a conductance and capacitance in parallel —denoted by $g_{i,k}$ and $C_{i,k}$ for the $i$-th branch incident to node $n_k$. Some elements may be missing, in which case the corresponding $G_{i,k}$ or $C_{i,k}$ is zero. The configuration covers generally all RC networks. However, no inductive elements are allowed.

The response of the central node when a step voltage is applied to the $i$-th terminal of it, all other terminals being grounded, is given by

$$h_{ik}(t) = \frac{G_{ik}}{\mathcal{G}_N} + \left(\frac{C_{ik}}{\mathcal{G}_N} - \frac{G_{ik}}{\mathcal{C}_N}\right) e^{t/\tau_k} \tag{3.1}$$

where

$$\mathcal{G}_N = \sum_{i=0}^{N-1} G_{ik} \quad ,\mathcal{C}_N = \sum_{i=0}^{N-1} C_{ik}, \quad \text{and} \quad \tau_k = \frac{\mathcal{C}_N}{\mathcal{G}_N}. \tag{3.2}$$

Since $\tau_k$ is in the form of time constant for general step responses of RC circuits, it is introduced by Sheehan as the *time constant of node* $n_k$ in the circuit. Since this time constant is independent of which neighbor or combination of neighbors is agitated, it is the characteristics of the node. nodes can be grouped in terms of their time constants: nodes with greatest and smallest time constants are called *slow* and *quick* nodes, respectively, and the others are called *normal* nodes. The classification can be quantified approximately by using some conversion of a range of time constants to a range of frequencies, e. g., $f = 2\pi/\tau$.

The importance of this classification comes from the fact that both quick and slow nodes can be eliminated from the network without significantly altering its behavior in the frequency range of interest. Beginning from the nodal equations of a RC network in $s$ domain:

$$(sM + G)\boldsymbol{x}(s) = P\boldsymbol{u}(s), \tag{3.3}$$

or

$$Y\boldsymbol{x}(s) = \boldsymbol{J}(s). \tag{3.4}$$



For simplicity, assume that the node we wish to eliminate is the last node $n_k$. Writing (3.4) as a block system

$$\begin{bmatrix} \tilde{Y} & \boldsymbol{Y} \\ \boldsymbol{Y}^T & \mathcal{G}_N + s\mathcal{C}_N \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{x}} \\ x_k \end{bmatrix} = \begin{bmatrix} \tilde{\boldsymbol{J}} \\ j_k \end{bmatrix}. \tag{3.5}$$

We can solve for $x_k$ from the second block equation and substitute it into the first block equation to obtain

$$\left(\tilde{Y} - E\right) x_k = \tilde{J} - \boldsymbol{F}, \tag{3.6}$$

where

$$E_{ij} = \frac{(G_{ik} + sC_{ik})(G_{jk} + sC_{jk})}{\mathcal{G}_N + s\mathcal{C}_N} \quad \text{and} \quad \boldsymbol{F}_i = \frac{G_{ik} + sC_{ik}}{\mathcal{G}_N + s\mathcal{C}_N} j_k. \tag{3.7}$$

In these equations $\mathcal{G}_N$ and $\mathcal{C}_N$ are defined analogously to the previous discussion, i. e.,

$$\mathcal{G}_N = \sum_{i=0}^{N-1} G_{ik} \quad \text{and} \quad \mathcal{C}_N = \sum_{i=0}^{N-1} C_{ik}. \tag{3.8}$$

Our goal is to realize $E_{ij}$ with positive RCLK elements. If we extend $E_{ij}$ in (3.7), we have

$$E_{ij} = \frac{G_{ik}G_{jk} + (G_{ik}C_{jk} + G_{jk}C_{ik})\,s + C_{ik}C_{jk}s^2}{\mathcal{G}_N + s\mathcal{C}_N}. \tag{3.9}$$

We now discuss the two extreme cases.

1. *Quick Nodes.* node $n_k$ is a quick node, i. e., $s\mathcal{C}_N \ll \mathcal{G}_N$. In this case, $\mathcal{G}_N + s\mathcal{C}_N \approx \mathcal{G}_N$. Therefore, we approximate element $E_{ij}$ from elimination by

$$E_{ij} \approx \frac{G_{ik}G_{jk}}{\mathcal{G}_N} + \frac{G_{ik}C_{jk} + G_{jk}C_{ik}}{\mathcal{G}_N} s + \frac{C_{ik}C_{jk}}{\mathcal{G}_N} s^2. \tag{3.10}$$

To realize it, the second-order term is simply neglected; hence,

$$E_{ij} \approx \frac{G_{ik}G_{jk}}{\mathcal{G}_N} + \frac{G_{ik}C_{jk} + G_{jk}C_{ik}}{\mathcal{G}_N} s. \tag{3.11}$$

The last equation can be translated into a procedure for physically modifying the circuit. To eliminate a quick node $n_k$ from a network, first remove all resistors and capacitors connecting other nodes to node $n_k$. Then insert new resistors and capacitors between former neighbors of $n_k$ according to the following two rules. If node $n_i$ and $n_j$ had been connected to $n_k$ through conductances $G_{ik}$ and $G_{jk}$, insert a conductance $G_{ik}G_{jk}/\mathcal{G}_k$; if node $n_i$ had a capacitor $C_{ik}$ to $n_k$ and $n_j$ had a conductance $G_{jk}$ to $n_k$, then insert capacitor of value $C_{ik}G_{jk}/\mathcal{G}_N$ between $n_i$ and $n_j$.

2. *Slow Nodes.* node $n_k$ is a slow node, i. e., $sC_N \gg \mathcal{G}_N$. In this case, $\mathcal{G}_N + sC_N \approx sC_N$. Therefore, we approximate element $E_{ij}$ in (3.9) by

$$E_{ij} \approx \frac{G_{ik}G_{jk}}{sC_N} + \frac{G_{ik}C_{jk} + G_{jk}C_{ik}}{C_N} + \frac{C_{ik}C_{jk}}{\mathcal{G}_N}s. \tag{3.12}$$

It is worth noting that, even though (3.12) can be realized by RLC in parallel, the to-be-realized circuit does not preserve the DC solution to the original one. This is because the 0-th order moment is not matched, i. e., $G_{ik}C_{jk} + G_{jk}C_{ik}/C_N \neq G_{ik}G_{jk}/\mathcal{G}_N$. To preserve DC characteristics, $G_{ik}G_{jk}/\mathcal{G}_N$ is used in place of whatever constant terms come from the expansion. In order to prevent causing ringing waveforms due to the co-existence of inductors and capacitors, $G_{ik}G_{jk}/sC_N$ is not included in the realization either; that is,

$$E_{ij} \approx \frac{G_{ik}C_{jk} + G_{jk}C_{ik}}{C_N} + \frac{C_{ik}C_{jk}}{\mathcal{G}_N}s. \tag{3.13}$$

From this we get the following slow-node elimination procedure. To eliminate a slow node $n_k$ from a network, first remove all resistors and capacitors connecting any nodes to node $n_k$. Then, as before, if nodes $n_i$ and $n_j$ had been connected to $n_k$ through conductances $G_{ik}$ and $G_{jk}$, insert conductance $G_{ik}G_{jk}/\mathcal{G}_k$ from $n_i$ to $n_j$; if node $n_i$ had a capacitor $C_{ik}$ to $n_k$, and node $n_j$ had a capacitor $C_{jk}$ to $n_k$, inert capacitor $C_{ik}C_{jk}/\mathcal{G}_k$ between $n_i$ and $n_j$.

### 3.1.1 Remarks

TICER employs Gauss elimination as the foundation of its node elimination strategies. And Gauss elimination is mathematically equivalent to Y-$\Delta$ transformation in graph theory. This topic will be fully discussed from Chapter 4.

The moments of $E_{ij}$ can be evaluated easily from (3.9):

$$m_0 = \frac{G_{ik}G_{jk}}{\mathcal{G}_N} \qquad m_1 = \frac{\mathcal{G}_N\left(G_iC_j + G_jC_i\right) - G_iG_jC_N}{\mathcal{G}_N}. \tag{3.14}$$

Generally, one can not guarantee that $m_1 \geq 0$. Therefore, TICER is not able to achieve the realizability and 1st-order moment matching simultaneously. As a direct result, the coefficient of $s$ for both the quick node and slow node elimination in (3.11) and (3.13) does not match $m_1$ in (3.14).

TICER's accuracy control is achieved by setting thresholds as the selection criteria for both quick and slow nodes. Plus, it is a first-order reduction method[4], so it is not devised to achieve high reduction ratio ($< 90\%$).

[4]The reduced models are 1st-order RC circuits.



## 3.2 Realizable RLC $\pi$-Model Reduction

Given a tree for which each branch is a RLC $\pi$ model, the two fundamental topologies within the tree is series connection (Fig. 3.7) and parallel connection (Fig. 3.8). There is a way to reduce the the original circuits in either topology.

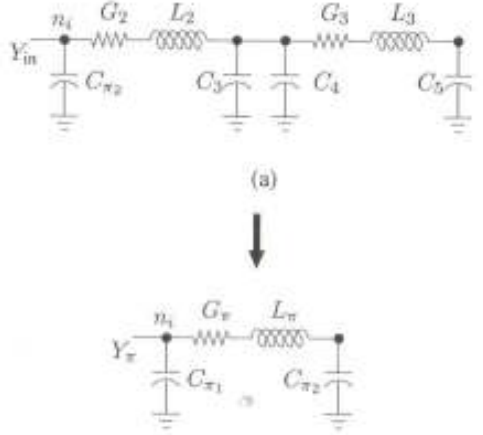



*Figure 3.7.* Two $\pi$ models connected in series. (a) The original circuit; (b) the reduced circuit.

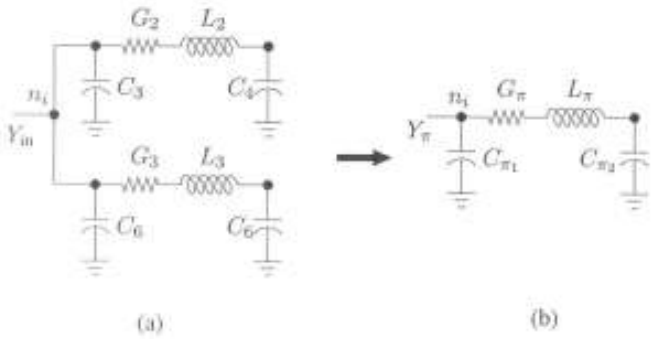



*Figure 3.8.* Two $\pi$ models connected in parallel. (a) The original circuit; (b) the reduced circuit.

The reduced models are realizable and kept in the $\pi$ structure. Furthermore, the driving-point admittances at the ports ($n_i$) of reduced circuits match the originals up to the 3rd order.

For the series topology, the driving-point admittance $Y_{in}$ at $n_i$ can be derived; i. e.,

$$Y_{in} = \frac{a_1 s + a_2 s^2 + a_3 s^3 + a_4 s^4 + a_5 s^5}{b_0 + b_1 s + b_2 s^2 + b_3 s^3 + b_4 s^4} = m_1 s + m_2 s^2 + m_3 s^3 + \cdots, \tag{3.15}$$

where

$$\begin{aligned}
a_1 &= G_2 G_3 (C_2 + C_3 + C_4 + C_5), \\
a_2 &= G_2 C_5 (C_2 + C_3 + C_4) + G_3 C_2 (C_3 + C_4 + C_5), \\
a_3 &= C_2 (C_3 + C_4) C_5 + G_2 G_3 (C_3 + C_4)(C_2 L_2 + C_3 L_3) \\
&\quad + G_2 G_3 C_2 C_5 (L_2 + L_3), \\
a_4 &= G_2 C_2 (C_3 + C_4) C_5 L_2 + G_3 C_2 (C_3 + C_4) C_5 L_3, \\
a_5 &= G_2 G_3 C_2 (C_3 + C_4) C_5 L_2 L_3, \\
b_0 &= G_2 G_3, \\
b_1 &= (G_2 + G_3) C_5 + G_3 (C_3 + C_4), \\
b_2 &= (C_3 + C_4) C_5 + G_2 G_3 (C_3 + C_4) L_2 + G_2 G_3 C_5 (L_2 + L_3), \\
b_3 &= (G_2 L_2 + G_3 L_3)(C_3 + C_4) C_5, \\
b_4 &= G_2 G_3 (C_3 + C_4) C_5 L_2 L_3.
\end{aligned} \tag{3.16}$$

After we get the coefficients, the next step is to assign values to the elements in Fig. (3.7):

$$C_{\pi_1} = \frac{a_2}{b_1} \qquad C_{\pi_2} = \frac{a_1 b_1 - a_0 b_2}{b_0 b_1} \tag{3.17}$$

$$G_\pi = \frac{b_1}{b_0 C_{\pi_2}} \qquad L_\pi = \frac{a_1 b_2 - a_3 b_0}{b_0^2 C_{\pi_2}^2}. \tag{3.18}$$

Since

$$\begin{aligned}
a_1 b_1 - a_0 b_2 &= G_2 G_3 \Big( (C_3 + C_4)^2 G_3 + 2 (C_3 + C_4) C_5 G_3 \\
&\quad + C_5^2 (G_2 + G_3) \Big) \\
a_1 b_2 - a_3 b_0 &= G_2 G_3 \Big( (C_3^2 + 2 C_4^2 + 2 C_3 C_4)(C_5 + G_2 G_3 L_2) \\
&\quad + (C_4 + C_3) C_5 (C_5 + 2 G_2 G_3 L_2) \\
&\quad + C_5^2 G_2 G_3 (L_2 + L_3) \Big),
\end{aligned}$$

by definition the reduced $\pi$ model is realizable. The driving-point admittance at $n_i$ in Fig. (3.7) is given by

$$Y_\pi = \frac{G_\pi (C_{\pi_1} + C_{\pi_2}) s + C_{\pi_1} C_{\pi_2} s^2 + G_\pi C_{\pi_1} C_{\pi_2} L_\pi s^3}{G_\pi + C_{\pi_2} s + G_\pi C_{\pi_2} L_\pi s^2}. \tag{3.19}$$



One can further verify, using the values given in (3.17)(3.18), that its first three moments match those of the original circuit, i. e., $m_1, m_2$ and $m_3$ in (3.15).

For the parallel topology, everything is the same as the scenario of the series topology, except that the coefficients (and thus moments) of the rational function in (3.15) are different from (3.16).

Equipped with these two kinds of realizable topological reduction, one can reduce a RLC $\pi$-modeled tree of any topology in the bottom-up fashion. Ultimately, the driving-point load is approximated by a single $RLC$ $\pi$-model. The response at the driving-point thus be evaluated using any gate delay calculators. However, to evaluate responses at any node in the tree, we have to evaluate transfer functions for each branch of the tree when doing the reduction, and propagate higher-level transfer functions all the way down to the nodes where responses are of interest.

This can be better explained using Fig. 3.9. Transfer functions from $n_i$ to $n_1$ and $n_2$ can be trivially obtained since $n_1$ and $n_2$ are leaf nodes. Suppose the

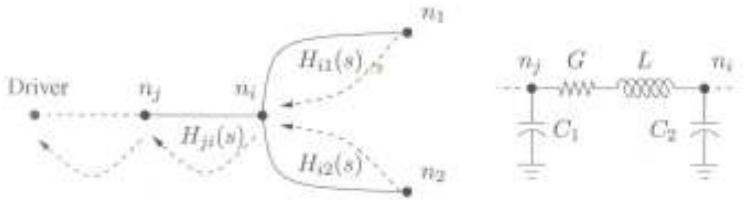



*Figure 3.9.* Transfer function evaluation and propagation. Each branch in the tree is a RLC $\pi$ model.

two transfer functions are available and denoted as $H_{i1}(s)$ and $H_{i2}(s)$. And the transfer function from $n_j$ to $n_i$ can be easily computed after the two downstream branches at $n_i$ are merged using the given procedure, i. e., it is to compute the response at the junction node of two series-connected $\pi$ models. Therefore, the computation of transfer functions of branches are conducted along the bottom-up reduction, and when $n_i$ is of interest, the transfer function from the driving node to $n_i$ is the product of all the transfer functions within the path from the driving node to $n_i$. Since the circuit is in tree structure, such a path is unique and guaranteed present.

**Lemma 3.1.** *Given two $k$-th order stable transfer functions $H_{ij}(s)$ and $H_{jk}(s)$, there is a $k$-th order stable transfer function $H_{ik}(s)$ which preserves the first $k$ moments of $H_{ij} H_{jk}$.*

We will discuss more on this in Chapter. 7. In the following we use a 3rd-order case as an example: suppose

$$H_{ij}(s) = \frac{1 + a_{j1}s + a_{j2}s^2}{1 + b_{j1}s + b_{j2}s^2 + b_{j3}s^3} \quad H_{jk}(s) = \frac{1 + a_{k1}s + a_{k2}s^2}{1 + b_{k1}s + b_{k2}s^2 + b_{k3}s^3},$$

are both stable, $H_{ij}H_{jk}$ can be approximated by (3.20), which is also stable and preserves the first three moments:

$$H_{ik}(s) = \frac{1 + a_{i1}s + a_{i2}s^2}{1 + b_{i1}s + b_{i2}s^2 + b_{i3}s^3}, \tag{3.20}$$

where

$$a_{i1} = a_{j1} + a_{k1} \qquad a_{i2} = a_{j2} + a_{k2} + a_{j1}a_{k1} \tag{3.21}$$
$$b_{i1} = b_{j1} + b_{k1} \qquad b_{i2} = b_{j2} + b_{k2} + b_{j1}b_{k1}. \tag{3.22}$$

### 3.2.1 Remarks

The realizable RLC $\pi$ model reduction was first proposed by Yang[102]. The method is able to achieve realizable reduction, and the reduced models are guaranteed stable. However, the method does have some limitations. The first limitation is on the geometry of the reducible circuits: it is only applicable to RLC $\pi$-modeled trees. Secondly, the realizable model has to be one-port only. To obtain responses at internal nodes of a tree, all we can get are reduced models (transfer functions), not realizable reduced circuits, which are more desirable.

## 3.3 Scattering-Parameter-Based Macro Model Reduction

### 3.3.1 What are Scattering Parameters?

To facilitate understanding of scattering parameters, we borrow an idea from billiards, or pool. One takes a cue ball and fires it up the table at a collection of other balls. After the impact, the energy and momentum in the cue ball is divided between all the balls involved in the impact. The cue ball *scatters* the stationary target balls and in turn is deflected or *scattered* by them.

In a distributed circuit, the equivalent to the energy and momentum of the cue ball is the amplitude and phase of the incoming wave on a port. This incoming wave is *scattered* by the circuit and its energy is partitioned between all the possible outgoing waves on all the other ports of the circuit.

**Definition 3.4 (Scattering Parameters).** Scattering parameters, which are commonly referred to as $S$-parameters, are a parameter set that relates those voltage waves $(a_i)$ scattered or reflected from the network to those voltage waves $(b_i)$ incident upon the network. Particularly for the 2-port network depicted in Fig. 3.10,

$$\begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}, \tag{3.23}$$



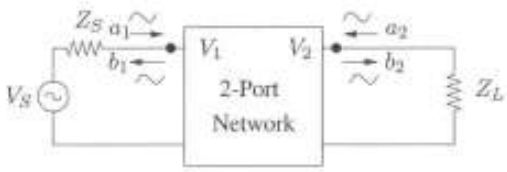



*Figure 3.10.* 2-port network showing incident waves $(a_1, a_2)$ and reflected waves $(b_1, b_2)$ used in scattering parameter definitions.

- $S_{11}$ is the reflection coefficient of the incident voltage wave $a_1$;
- $S_{22}$ is the reflection coefficient of the incident voltage wave $a_2$;
- $S_{12}$ is the transmission gain from the incident voltage wave $a_2$ to the left port;
- $S_{21}$ is the transmission gain from the incident voltage wave $a_1$ to the right port.

In addition, the total voltage waves at the two ports are

$$V_1 = a_1 + b_1 \tag{3.24}$$
$$V_2 = a_2 + b_2, \tag{3.25}$$

and the currents at the ports are defined as

$$I_1 = \frac{(a_1 - b_1)}{Z_{c_1}} \tag{3.26}$$
$$I_2 = \frac{(a_2 - b_2)}{Z_{c_2}}, \tag{3.27}$$

where $Z_{c_i}$ is the characteristic impedances.

The definition can be easily extended for $n$-port network $(n > 2)$.

Suppose a network has $n$ ports, then the $S$-parameters of the network will be a $n \times n$ matrix. According to the definition,

$$S_{ij} = \left\{ \frac{b_i}{a_j}\Big|_{a_k=0}, \text{ for } k = 1, 2, \ldots, n \text{ and } k \neq j \right\} \tag{3.28}$$

we connect all ports by resistors whose resistance are equal to their respective characteristic impedances, which makes $a_k = 0$ for $k = 1, 2, \ldots, n$. Then set initial incident wave at port $j$ be unity, i. e., $a_j = 1$, and measure reflective waves at all port $b_i$ for $i = 1, 2, \ldots, n$. From above definition, we have $S_{ij} = b_i$. In this way, we will measure all the $S$-parameters of the network.

So far, we have introduced $S$-parameters and admittance matrix ($Y$-parameters) of linear multi-port networks. Note that we have used different sets of independent and dependent variables for the two kinds of parameters[5], However, all parameter sets contain the same information about a network, and it is always possible to calculate any set in term of any other set. For example, $S$-parameters can be written in terms of $Y$-parameters as

$$S(s) = (I + Y)^{-1}(I - Y). \tag{3.29}$$

And alternatively, $Y$-parameters can be represented by $S$-parameters:

$$Y(s) = (I + S)^{-1}(I - S). \tag{3.30}$$

### 3.3.2 Scattering-Parameter-Based Reduction

We would derive the reduction merely based on $S$-matrix first. And we will explain the result using the original multi-port network on which the $S$-matrix is defined. Let us start with the most general case in (3.31), where each entry in the $S$-parameter matrix is a symbol (full matrix). We want to eliminate two independent variable $a_1$ and $a_2$ using Gauss elimination.

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{bmatrix} = \begin{bmatrix} S_{11} & S_{12} & S_{13} & \cdots & S_{1n} \\ S_{21} & S_{22} & S_{23} & \cdots & S_{2n} \\ S_{31} & S_{32} & S_{33} & \cdots & S_{3n} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ S_{n1} & S_{n2} & S_{n3} & \cdots & S_{nn} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_n \end{bmatrix}. \tag{3.31}$$

From the first equation of (3.31),

$$a_1 = \frac{b_1 - S_{12}a_2 - S_{13}a_3 - \cdots - S_{1n}a_n}{S_{11}}. \tag{3.32}$$

Replace $a_1$ using the above equation, the last $n-1$ equations in (3.31) can be rewritten as

$$\begin{bmatrix} b_2 - \frac{b_1 S_{21}}{S_{11}} \\ b_3 - \frac{b_1 S_{31}}{S_{11}} \\ \vdots \\ b_n - \frac{b_1 S_{n1}}{S_{11}} \end{bmatrix} = \begin{bmatrix} S_{22} - \frac{S_{12}S_{21}}{S_{11}} & S_{23} - \frac{S_{13}S_{21}}{S_{11}} & \cdots & S_{2n} - \frac{S_{1n}S_{21}}{S_{11}} \\ S_{32} - \frac{S_{12}S_{31}}{S_{11}} & S_{33} - \frac{S_{13}S_{31}}{S_{11}} & \cdots & S_{3n} - \frac{S_{1n}S_{31}}{S_{11}} \\ \cdots & \cdots & \cdots & \cdots \\ S_{n2} - \frac{S_{12}S_{n1}}{S_{11}} & S_{n3} - \frac{S_{13}S_{n1}}{S_{11}} & \cdots & S_{nn} - \frac{S_{1n}S_{n1}}{S_{11}} \end{bmatrix} \begin{bmatrix} a_2 \\ a_3 \\ \vdots \\ a_n \end{bmatrix} \tag{3.33}$$

[5] We have used incident waves and reflective waves as independent and dependent variables in $S$-parameters, and nodal voltages and branch currents as independent and dependent variables in $Y$-parameters, respectively.



Now we go through another similar iteration to eliminate $a_2$ in (3.33); we have

$$\begin{bmatrix} \tilde{b}_3 \\ \vdots \\ \tilde{b}_n \end{bmatrix} = \begin{bmatrix} \tilde{S}_{33} & \cdots & \tilde{S}_{3n} \\ \cdots & \cdots & \cdots \\ \tilde{S}_{n3} & \cdots & \tilde{S}_{nn} \end{bmatrix} \begin{bmatrix} a_3 \\ \vdots \\ a_n \end{bmatrix}, \tag{3.34}$$

where

$$\tilde{b}_i = b_i - \frac{b_1 S_{i1}}{S_{11}} - \left(b_2 - \frac{b_1 S_{21}}{S_{11}}\right) \frac{S_{i1}}{S_{11}S_{22} - S_{12}S_{21}} \tag{3.35}$$

and

$$\tilde{S}_{ij} = S_{ij} + \frac{-S_{i2}S_{2j}S_{11} + S_{i2}S_{21}S_{1j} + S_{i1}S_{12}S_{2j} - S_{i1}S_{1j}S_{22}}{S_{11}S_{22} - S_{12}S_{21}}. \tag{3.36}$$

In (3.35), although the matrix size has be reduced by 2, $\tilde{b}_i$ is still related to $b_1$ and $b_2$. As $b_1$ and $b_2$ are used to represent $a_1$ and $a_2$, it is impossible to eliminate $a_1$ and $a_2$ without introducing $b_1$ and $b_2$ into the reduced system, except that $b_1$ and $b_2$ are zero.

We can continue the elimination process to further reduce the size of $S$-matrix. But let us stop here and turn to look at the physical meaning of eliminating two nodes in $S$-matrix from the circuit point of view.

The network reduction problem based on $S$-parameters can be defined as follows: given a linear distributed-lumped network, find a multiport representation of the network as illustrated by Fig. 3.11, where the multiport is characterized by its $S$-matrix. All nodes in the network are internal to the multiport except the node connected to the driving source ($n_1$) and the loads of interest ($n_2$ through $n_m$). These external nodes are specified by the user.

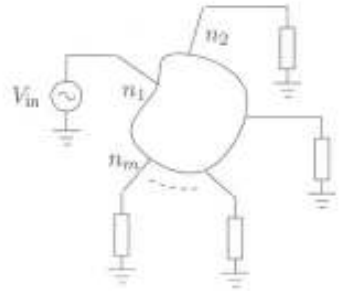



*Figure 3.11.* A multiport representation

To obtain such a multiport representation with $m$ external ports from an arbitrary distributed-lumped network of $n$ original nodes, the network is reduced

by merging the nodes into the multiport one at a time while keeping all user specified nodes external. There are two basic reduction rules:

**Adjoined Merging Rule:**

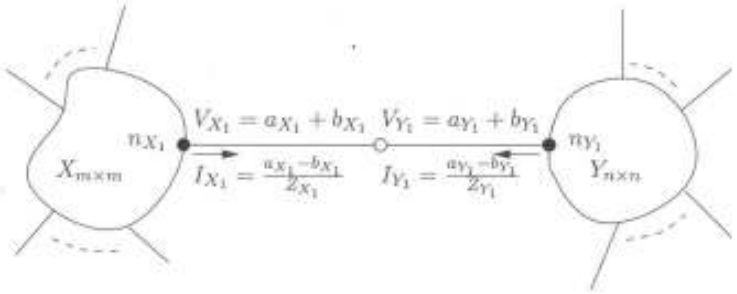



*Figure 3.12.* Merge the two networks denoted by $X_{m\times m}$ and $Y_{n\times n}$ at two perfectly interconnected nodes $n_{X_1}$ and $n_{Y_1}$.

Let $X$ and $Y$ be two multiport network in Fig. 3.12. If the two networks share the same ground, but are *not* connected at $n_{X_1}$ and $n_{Y_1}$, the $S$-matrix for the two networks are given by

$$b = \begin{bmatrix} S_{X_1X_1} & S_{X_1X_2} & \cdots & S_{X_1X_m} & 0 & 0 & \cdots & 0 \\ S_{X_2X_1} & S_{X_2X_2} & \cdots & S_{X_2X_m} & 0 & 0 & \cdots & 0 \\ \hline S_{X_mX_1} & S_{X_1X_2} & \cdots & S_{X_mX_m} & 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & S_{Y_1Y_1} & S_{Y_1Y_2} & \cdots & S_{Y_1Y_n} \\ 0 & 0 & \cdots & 0 & S_{Y_2Y_1} & S_{Y_2Y_2} & \cdots & S_{Y_2Y_n} \\ \hline 0 & 0 & \cdots & 0 & S_{Y_nY_1} & S_{Y_nY_2} & \cdots & S_{Y_nY_n} \end{bmatrix} a \tag{3.37}$$

in which,

$$\begin{aligned} a &\equiv \begin{bmatrix} a_{X_1} & a_{X_2} & \cdots & a_{X_m} & a_{Y_2} & a_{Y_2} & \cdots & a_{Y_n} \end{bmatrix}^T, \\ b &\equiv \begin{bmatrix} b_{X_1} & b_{X_2} & \cdots & b_{X_m} & b_{Y_1} & b_{Y_2} & \cdots & b_{Y_n} \end{bmatrix}^T, \\ S_{ij} &= 0 \text{ if } n_i \in X \text{ and } n_j \in Y. \end{aligned} \tag{3.38}$$

Now if the two networks are perfectly interconnected at $n_{X_1}$ and $n_{Y_1}$, then the voltages at the two nodes are equal; and for the central node in between, KCL holds. Therefore, besides (3.37), we have two additional equations, i. e.,

$$a_{X_1} + b_{X_1} - (a_{Y_1} + b_{Y_1}) = 0 \tag{3.39}$$

$$\frac{a_{X_1} - b_{X_1}}{Z_{X_1}} + \frac{a_{Y_1} - b_{Y_1}}{Z_{Y_1}} = 0 \tag{3.40}$$



or

$$\begin{bmatrix} b_{X_1} \\ b_{Y_1} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} a_{X_1} \\ a_{Y_1} \end{bmatrix}, \tag{3.41}$$

because $n_{X_1}$ and $n_{Y_1}$ are actually the same node, so $Z_{X_1} = Z_{Y_1}$.

Insert (3.41) into (3.37) by replacing $b_{X_1}$ and $b_{X_2}$, we have

$$b' = \begin{bmatrix} S_{X_1X_1} & S_{X_1X_2} & \cdots & S_{X_1X_m} & \boxed{-1} & 0 & \cdots & 0 \\ S_{X_2X_1} & S_{X_2X_2} & \cdots & S_{X_2X_m} & 0 & 0 & \cdots & 0 \\ \hline S_{X_mX_1} & S_{X_1X_2} & \cdots & S_{X_mX_m} & 0 & 0 & \cdots & 0 \\ \boxed{-1} & 0 & \cdots & 0 & S_{Y_1Y_1} & S_{Y_1Y_2} & \cdots & S_{Y_1Y_n} \\ 0 & 0 & \cdots & 0 & S_{Y_2Y_1} & S_{Y_2Y_2} & \cdots & S_{Y_2Y_n} \\ \hline 0 & 0 & \cdots & 0 & S_{Y_nY_1} & S_{Y_nY_2} & \cdots & S_{Y_nY_n} \end{bmatrix} a \tag{3.42}$$

where

$$b' \equiv \begin{bmatrix} \boxed{0} & b_{X_2} & \cdots & b_{X_m} & \boxed{0} & b_{Y_2} & \cdots & b_{Y_n} \end{bmatrix}^T.$$

Note that in (3.42 only boxed entries are changed.

Comparing (3.31) with the above equation, if we eliminate $n_{X_1}$ and $n_{Y_1}$, it is equivalent to eliminate the two rows with $-1$ in (3.42). Because of some special values (e. g., 0, $-1$) in the above equation, we could rewrite (3.35) and (3.36),

$$\tilde{b}_i = b_i \tag{3.43}$$

$$\tilde{S}_{ij} = \begin{cases} S_{ij} + \frac{-S_{iX_1}S_{X_1j}S_{Y_1Y_1}}{S_{X_1X_1}S_{Y_1Y_1}-1} & i,j \in X \\ S_{ij} + \frac{-S_{iY_1}S_{Y_1j}S_{X_1X_1}}{S_{X_1X_1}S_{Y_1Y_1}-1} & i,j \in Y \\ -\frac{S_{iX_1}S_{Y_1j}}{S_{X_1X_1}S_{Y_1Y_1}-1} & i \in X, j \in Y \\ -\frac{S_{iY_1}S_{X_1j}}{S_{X_1X_1}S_{Y_1Y_1}-1} & i \in Y, j \in X \end{cases} \tag{3.44}$$

Note that (3.43) is derived because of the two zeros in the left column in (3.42).

**Self Merging Rule:**

Let $X$ be an $m$-port network with a self loop connected to $n_{X_1}$ and $n_{X_2}$ in $X$ (see Fig. 3.13). The only difference of the self-merging scenario from adjoined-merging is that the equation (3.38) can not be applied in self-merging rule. However, (3.41) still holds when $Y_1$ is replaced by $X_2$, i. e.,

$$\begin{bmatrix} \boxed{0} \\ \boxed{0} \\ b_{X_3} \\ \vdots \\ b_{X_m} \end{bmatrix} = \begin{bmatrix} S_{X_1X_1} & S_{X_1X_2}\boxed{-1} & S_{X_1X_3} & \cdots & S_{X_1X_m} \\ S_{X_2X_1}\boxed{-1} & S_{X_2X_2} & S_{X_2X_3} & \cdots & S_{X_2X_m} \\ \hline S_{X_mX_1} & S_{X_1X_2} & S_{X_mX_3} & \cdots & S_{X_mX_m} \end{bmatrix} \begin{bmatrix} a_{X_1} \\ a_{X_2} \\ a_{X_3} \\ \vdots \\ a_{X_m} \end{bmatrix}. \tag{3.45}$$

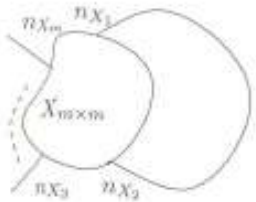



*Figure 3.13.* Illustration of self merging

Therefore, if we eliminate the first two rows from the above system, then the entries are given by

$$\tilde{b}_i = b_i \tag{3.46}$$

$$\tilde{S}_{ij} = S_{ij} + \frac{S_{iX_2}\left(\left(S_{X_2X_1}-1\right)S_{X_1j} - S_{X_2j}S_{X_1X_1}\right)}{\Delta} \tag{3.47}$$

$$+\frac{S_{iX_1}\left(\left(S_{X_1X_2}-1\right)S_{X_2j} - S_{X_1j}S_{X_2X_2}\right)}{\Delta}, \quad i,j = 1,\cdots,m-2. \tag{3.48}$$

where

$$\Delta = S_{X_1X_1}S_{X_2X_2} - \left(\left(S_{X_1X_2}-1\right)\left(S_{X_2X_1}-1\right)\right). \tag{3.49}$$

Given an arbitrary distributed-lumped network, let $E_n$ be the set of external $n$ nodes. The network reduction process begins with merging all internal components by repeatedly utilizing the adjoined merging rule for all the nodes $n_i$ that does not belong to $E_n$. The self merging rule is applied to eliminate all the self loops introduced by the adjoined merging process. Finally, an $n$-port network characterized by its scattering parameters is derived. Note that the $S$-parameters are approximated by their lower order moments.

### 3.3.3 Getting Transfer Functions

Once we have obtained the reduced $S$-matrix, we can use (3.25) and (3.27) to covert $a_i$ and $b_i$ into port voltages and currents. Thus, we can use various combinations of them to get any transfer function of interest. Once the transfer function is obtained, the Padé approximation method introduced before can be used to analyze the system.

### 3.3.4 Remarks

$S$-parameter based macro model of distributed-lumped networks was first introduced by Liao[51]. The S-parameter based macromodel is flexible that the accuracy of the model can be controlled by adjusting the order of approximation. However, it uses Padé approximation to obtain macromodels. Later on, Liao proposed a realizable reduction method[50] based on $S$-parameters. Yet the method is only applicable to RC circuits only, and realizable circuit is first order only. Another limitation of the proposed macromodel method is that scattering parameters are used only as an intermediate result, as the given distributed-lumped networks and desired transfer functions (macromodels) are all in Laplace transform. Therefore, it is apparently more preferable to use Laplace transforms directly. Generalized Y-$\Delta$ transformation is just one such method (Chapter 4).



## 4. Summary

In this chapter, we first laid the foundation for linear circuit simulation and reduction —basics of circuit analysis in time domain and $s$-domain. We then presented two state-of-the-art research directions in linear reduction area: explicit moment-matching method with Padé approximation and realizable topological reduction methods. The two directions have their advantages and limitations. Nowadays, these the methods are all implemented and widely used to solve real industry designs. And their limitations, however, are the motivations of the research in this thesis —generalized Y-$\Delta$ transformation.