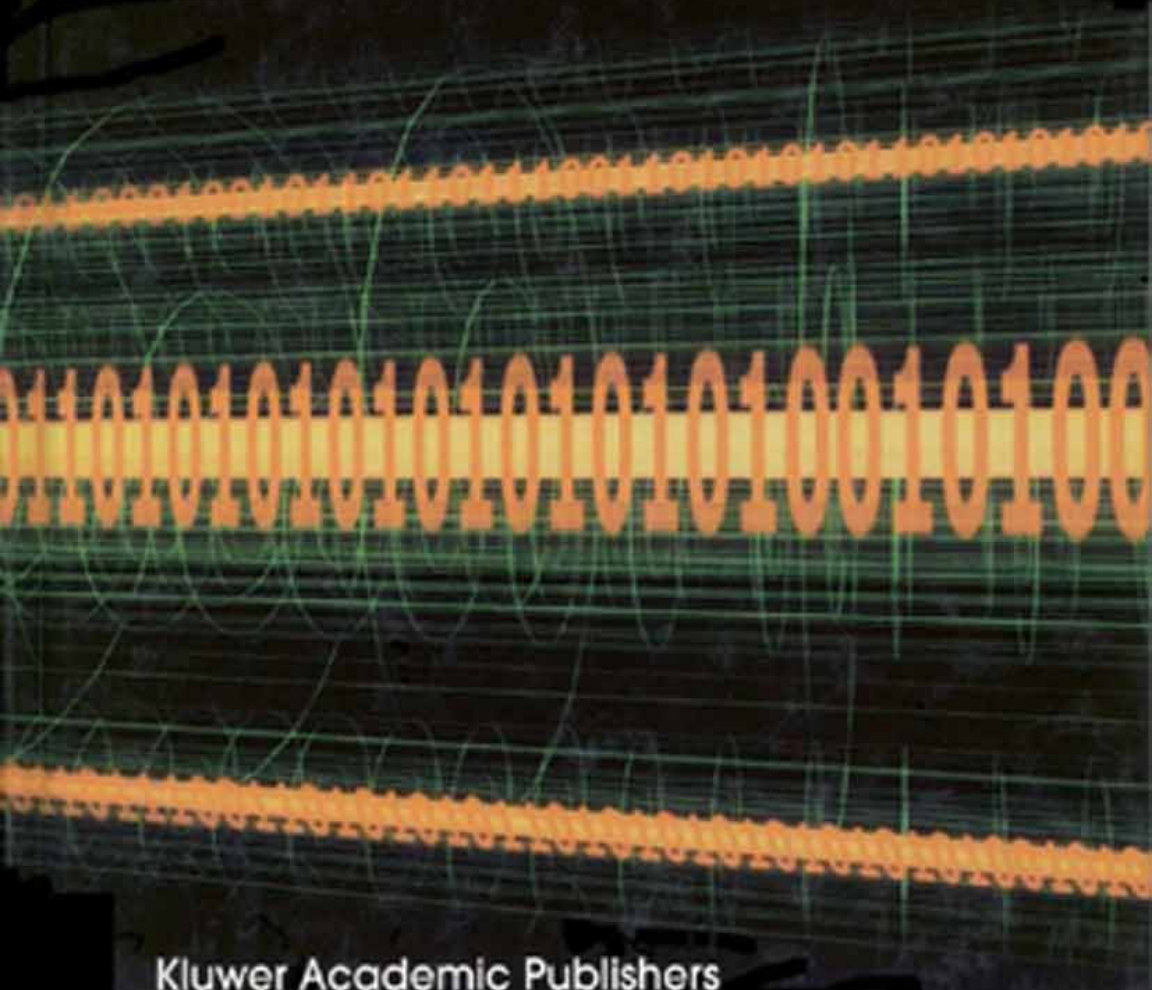
IC Interconnect Analysis

Mustafa Celik
Lawrence Pileggi
Altan Odabasioglu



Kluwer Academic Publishers

IC Interconnect Analysis

IC Interconnect Analysis

Mustafa Celik
Monterey Design Systems

Lawrence Pileggi
Carnegie Mellon University

Altan Odabasioglu
Avant! Corporation

KLUWER ACADEMIC PUBLISHERS
NEW YORK, BOSTON, DORDRECHT, LONDON, MOSCOW

eBook ISBN: 0-306-47971-0
Print ISBN: 1-4020-7075-6

Created in the United States of America

Visit Kluwer Online at: http://kluweronline.com
and Kluwer's eBookstore at: http://ebooks.kluweronline.com

Contents

Preface ..... **ix**

1 Introduction **1**

1.1 Interconnect Trends 1
1.2 Interconnect Models 6
1.3 Interconnect Analysis Via Moments 12
1.4 Interconnect Metrics 13
1.5 Moment Matching and Model Order Reduction 17
1.6 Summary 20

2 The Elmore Delay **25**

2.1 Delay of a Transfer Function 25
2.2 RC Interconnect Delay 31
2.3 The Elmore Delay 33
2.4 Moments 39
2.5 Extending Elmore's Distribution Theory Analogy 44
2.6 The Elmore Delay as a Bound 46
2.7 The Elmore Delay for General Input Signals 49
2.8 The Elmore Delay as a Bound for Phase Delay In RC Trees 54
2.9 Summary 57
2.A Appendix 57

3 Higher-Order RC(L) Delay Metrics **67**

3.1 Gamma Distribution Model 67
3.2 h-gamma: Gamma Fitting of Homogeneous Response 73
3.3 Double Exponential Impulse Response Distribution 80
3.4 Closed Form RC Delay Metrics 94
3.5 Noise Metrics 95
3.6 RLC Interconnect 100
3.7 Signal Attenuation and Phase Delay in RLC Trees 105
3.8 Summary 113

3.A Appendix 113

4 Asymptotic Waveform Evaluation 119

4.1 State Equations and Circuit-Response Functions 119
4.2 Moments 121
4.3 Moment Matching 126
4.4 Practical Considerations 129
4.5 Padé Approximation 132
4.6 Moment-Matching Issues 137
4.7 Multi-point AWE Methods 149
4.8 Summary 150

5 Moment Generation 153

5.1 Calculating Moments in Tree-Like Circuits 153
5.2 Calculating Moments Using MNA 170
5.3 Calculating Moments in Transmission Line Circuits 177
5.4 Summary 179

6 Passive Reduced-Order Multiport Models 181

6.1 Multiport Modeling 181
6.2 Macromodeling Using AWE 189
6.3 Krylov Subspaces 191
6.4 Projection Methods for Order Reduction 194
6.5 Stability and Passivity 202
6.6 PRIMA 204
6.7 Practical Issues 218
6.8 Special Cases in Multiport Modeling 228
6.9 Summary 234
6.A Appendix -- Proofs of the Lemmas 234
6.B Appendix -- Error Estimation in PRIMA 236

7 Interfacing with SPICE 243

7.1 Multiport Interconnect Models 243
7.2 Post Processing of the PRIMA Models 245

7.3 State-Space Realization from Poles and Residues 251
7.4 Synthesis of State-Space Realizations 256
7.5 Recursive Convolution 258
7.6 Summary 268

8 Interfacing Interconnect and Gate-Delay Models 271

8.1 Logic Stage Delay Calculation 271
8.2 Gate Characterization 272
8.3 Effective Capacitance Concept 275
8.4 Two-Step Delay Approximation 276
8.5 Thevenin Delay Modeling 277
8.6 Thevenin Delay Model Computation Details 282
8.7 Gate Models for General RLC Loading 289
8.8 Interconnect Coupling 292
8.9 Summary 305

Index 307

Preface

The advent of sub-quarter-micron IC technologies has forced dramatic changes in the design and manufacturing methodologies for integrated circuits and systems. The paradigm shift for interconnect -- which was once considered just parasitic but can now be a dominant factor for integrated circuit performance -- provided the greatest impetus for change of existing methodologies. Over the past decade there have been a number of advances in modeling and analysis of interconnect that have facilitated the continual advances in design automation for systems of increasing size and frequency. This book provides a comprehensive coverage of modeling and simulation of RC and RCL interconnect, including the interactions with gate- and transistor-level models, such as those used in SPICE. The practical aspects of the algorithms and models are explained with sufficient detail for the interested practitioner, but also include some theoretical depth for those who wish to look beyond these state-of-the-art methods.

This book is primarily a compilation of research that was carried out at the University of Texas at Austin, Carnegie Mellon University, and Monterey Design Systems. We would like to acknowledge the contributions from several colleagues and friends to various chapters throughout this book. We especially want to thank Emrah Acar, Ravi Arunachalam, Florin Dartu, Rohini Gupta, Byron Krauter, Tao Lin, Frank Liu, Noel Menezes, Satya Pullela, Jessica Qian, Curtis Ratzlaff, and Bogdan Tutuianu for contributions of excerpts from their published papers. We wish to especially thank Xin Li, Rusen Oktem, and Guy Maor for their outstanding technical review of the chapters and formulas. We would also like to thank Patricia Hermenault for her grammatical editing, and Elizabeth Miller for the design of the book cover. Special thanks also to Carl Harris at Kluwer Academic Publishers for soliciting the proposal for this book and then enthusiastically encouraging us to complete it.

This book would never have been completed without support from many sources. Mustafa Celik would like to acknowledge the support of Tak Young and Monterey Design Systems for giving him the time to work on this project. And most importantly, we are deeply indebted to the support that we received from our families: Mustafa's wife Asli and son Kerem, and Larry's wife Leah and daughter Hannah, who support us in everything that we do.

Mustafa Celik
Larry Pileggi
Altan Odabasioglu

CHAPTER 1

Introduction

As integrated circuit feature sizes continue to scale well below 0.25 microns, active-device counts are reaching hundreds of millions. The amount of interconnect among the devices tends to grow superlinearly with the transistor counts, and the chip area is often limited by the physical interconnect area. Due to these interconnect area limitations, the interconnect dimensions are scaled with the devices whenever possible [1.1] [1.2]. In addition, to provide more wiring resources, IC's now accommodate numerous metallization layers, with more to come in the future [1.3].

These advances in technology that result in scaled, multi-level interconnects may address the wireability problem, but in the process create problems with signal integrity and interconnect delay. We introduce some of these interconnect problems in the remainder of this chapter, then the seven chapters which follow describe in detail various solutions for these problems.

1.1 Interconnect Trends

At maximum wiring density, each IC wire is electrostatically coupled to nearest neighbors on the same layer, as well as wires above and below it, as sketched in Figure 1.1. If all of the dimensions (including conductor thicknesses) in Figure 1.1 are scaled by *S*, the capacitance per unit length (cross-section) among the wires

remains unchanged. In contrast, the resistance per unit length for each wire is increased by $1/S^2$. Therefore, the RC per unit length is increased by the same factor.

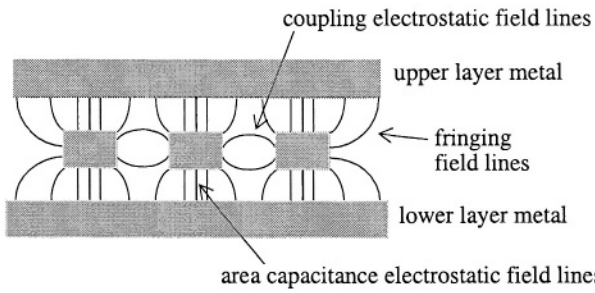


FIGURE 1.1 Multi-level metal interconnect cross-section. Wires are alternated in direction from layer-to layer.

If all of the lengths of interconnect are scaled -- which would be the case for a complete die shrink -- then the total RC for the interconnects would remain unchanged. But as device sizes are reduced, there has been a tendency to place more functionality on the chip, and therefore the average interconnect lengths do not scale. Moreover, as the devices are scaled, there may be an improvement in their operating speed or output impedance that will make the R of the interconnect relatively larger in comparison to the "R" of the driver. So, while the operating speed of the gates is improving, the delays of the interconnect between the gates remain fixed. This can translate into an increase in the interconnect delay, relative to the driver delay, even when the lengths are scaled.

But this analysis assumes that the conductor thicknesses are scaled, and this is not always desirable if interconnect resistance is required to be smaller. If the conductor thicknesses are held constant, then R per unit length of interconnect increases by only 1/*S*. However, due to fringing and coupling capacitance effects, the capacitance per unit length increases when conductor thicknesses are not scaled, and the RC per unit length increase is greater than 1/*S*.

1.1.1 Local Wires and Global Wires

When technology trends are investigated, usually an important distinction is made between two types of wires. *Local wires* connect gates or cells within blocks, and with scaling these wires get shorter. *Global wires* connect blocks together and usually

span a significant portion of a chip. They do not shrink when devices get smaller, and even tend to increase with increasing chip sizes.

Resistance does not matter much for local wiring since the total resistance is dominated by the device resistance. Global wires, on the other hand, are most impacted by RC effects. This is due to the combined effect of decreasing driver resistance to drive the large load and large wire resistances due to the long length. The actual lengths, hence the metal resistance impact, also depend on how global wires are buffered.

Local wires do not have significant RC interconnect delays, but their total capacitance can be dominated by the interconnect capacitance since wire capacitance does not decrease with scaling, whereas device capacitance does. Furthermore, when C dominates, coupling can be a big factor, and in effect, it can scale the capacitances several times their nominal value. This affects both local and global nets.

1.1.2 Coupling Capacitance Effects

When the conductor spacings become comparable to the conductor thicknesses, the coupling capacitance between wires is significant (refer to Figure 1.2). One has to be concerned with the coupling between signal wires, as it impacts performance and signal integrity. Moreover, assuming that the wiring layers in Figure 1.2 represent upper and lower level metal layers, it should be noted that the majority of the total capacitance will be between signal wires for multi-level technologies, and very little capacitance will be to the substrate (ground).

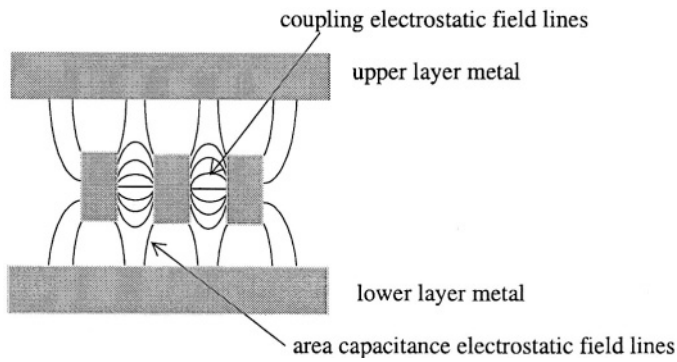


FIGURE 1.2 In DSM IC's wire capacitance is dominated by the coupling capacitance between the same layer wires.

For delay analysis purposes we would like to create simple RC models with the capacitance connected from the line to ground. Representing all of the coupling to other wires is intractable for many applications. But if the coupling is to be modeled as a grounded C, the value should be adjusted to consider the worst case conditions of switching from the other line(s).

Consider two coupled lines, as shown in Figure 1.3. If one line is switching high, while the other is switching low, the waveform on line 1 may become non-monotone, and the "delay" is increased. If, for delay calculation purposes, we want to analyze the delay of line 1, independent of line 2, then we can consider modeling the coupling capacitance by an effective capacitance to ground.

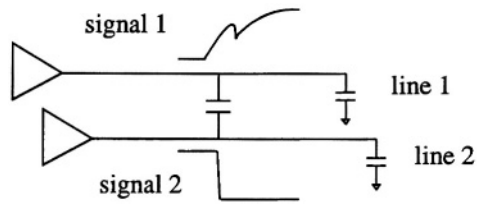


FIGURE 1.3 Two coupled lines.

For example, referring to Figure 1.4, assume that the rate of change of signal 1 is $\dot{v}(t)$ and that of signal 2 is $\beta\dot{v}(t)$ (where $\beta \geq 1$) during the time period of switching for line 1, ΔT. While line 2 is switching, the current through the coupling capacitor is $C(1+\beta)\ \dot{v}(t)$, for a time duration of VDD/β. The effective capacitance for line 1 (if the coupling capacitor is modeled as a C to ground) is $C(1+\beta)$ for this period of time. For the remaining portions of the switching period on line 1, the capacitance is approximately C (it is actually slightly less than C since line 2 is not a perfect ground) for the duration of $\Delta T - VDD/\beta$.

In summary, we have a brief period of time during which the effective capacitance is quite large, and then the remaining time the effective capacitance is approximately C. One can average these two capacitors, over the corresponding time periods, which results in an average effective capacitance of $2C$ (as expected since the total change in voltage is $2VDD$). But the actual effect on delay is obtained only by a complete simulation of the coupled lines. The impact on delay can be significantly greater than that modeled by an effective capacitor of 2C.

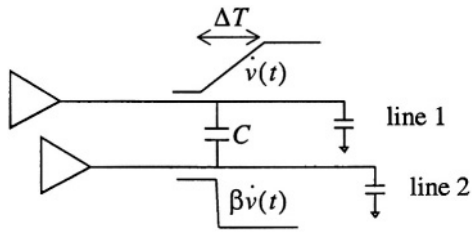


FIGURE 1.4 Ideal waveshape assumption.

If the lines are switching in the same direction, one can derive a similar formula for the *decrease* in effective capacitance. Since most of the coupling is between signal lines, the pessimistic/optimistic impact of coupling on delay is extremely important.

1.1.3 Inductance Effects

At extremely fast switching frequencies there is the potential for inductance to play a role in the calculation of interconnect delay. This largely depends on the design style; particularly the length of the unbuffered wires and the power/ground line configurations for returning current.

As chip sizes grow, it is increasingly difficult to run long, resistive, metal lines across the chip and operate at or near gigahertz frequencies. For this reason there have been proposals for using thicker, wider, hence lower resistance metal for the top layer of interconnect to reduce the RC delay. However, if the R per unit length is reduced significantly, the inductance can become a factor. In general, on-chip inductance is most likely to be evident in large clock distribution trunks and wide signal busses. For busses, the aggregate effect of many parallel wires behaves similarly to one large wire.

Perhaps more importantly, as the boundary between chip and package becomes more important (with flip-chip technologies, etc.), the inductance of the package and chip-to-package interface could become extremely critical for determining the overall IC system performance prior to manufacturing. For this reason, effective inductance modeling and analysis is required for any high performance IC design.

1.2 Interconnect Models

This book does not describe the algorithms used to generate resistance, capacitance, and inductance values from interconnect geometry information, but below we include a brief introduction to this subject and describe the circuit models that they produce.

1.2.1 RC Trees

Most interconnect model topologies follow a tree like structure as shown in Figure 1.5(a). For such a structure, the simplest electrical model that includes the topology information and metal resistivity is in the form of an RC tree (see Figure 1.5(b)). An RC tree is an RC circuit with capacitors from all nodes to ground, no capacitors between nonground nodes, and no resistors connected to ground. To have such a simple model, drivers are sometimes modeled in the form of a Thevenin equivalent voltage source, and the receivers are replaced by linear capacitors.

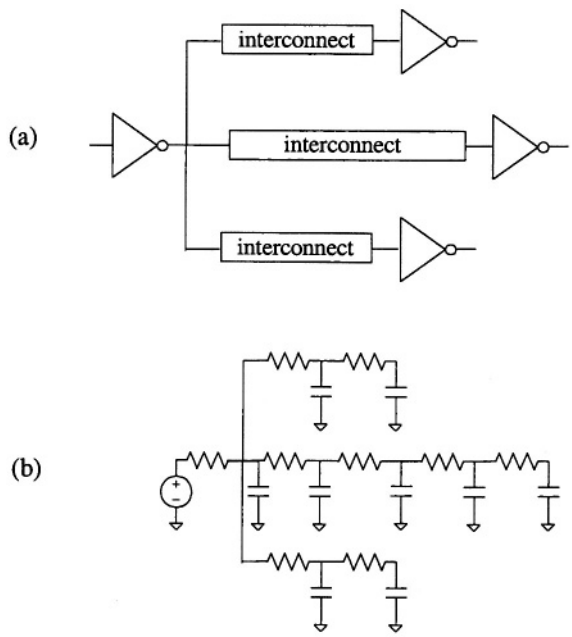


FIGURE 1.5 An interconnect and its electrical model.

The resistance calculation for interconnect is straightforward. For a uniform structure with a rectangle cross-section the resistance is given by

$$R = R_s\left(\frac{l}{w}\right) \tag{1.1}$$

where $R_s = \rho/t$ is the process dependent sheet resistance having units of Ω/square, and ρ, l, t, and w are the resistivity, length, thickness, and width of the wire, respectively. Thus to obtain the resistance of a wire on a layer we simply multiply the sheet resistance by the ratio of the length to width of the wire.

For the nonuniform or nonrectangle structures such as vias and corners, the resistance calculations are more complicated than (1.1). One approach can be to break the conductor into simple regions so that (1.1) can be used for each region. A more straightforward and accurate method is to formulate the problem in terms of the 2-D or 3-D Laplace equations and solve them via finite-difference methods.

For capacitance extraction many different techniques can be employed. Depending on the desired accuracy, these methods can vary from using very simple 2-D analytical models to employing 3-D electrostatic field solvers.

For example, the empirical formula [1.4]

$$C = \varepsilon\left[1.15\left(\frac{w}{h}\right) + 2.8\left(\frac{t}{h}\right)^{0.222}\right] \tag{1.2}$$

and its derivatives were very useful to estimate the per-unit-length wire capacitance, when the wires in a chip could be modeled as a single conductor over a ground plane.

But with multiple routing layers and increasing routing density, the capacitance interactions between conductors can become very complex, and the closed-form expressions such as (1.2) can no longer be used. In deep submicron IC design methodologies, the generally applied approach is to precharacterize the capacitance values for the commonly encountered 2-D and/or 3-D structures. The exact capacitance values for these structures are calculated *a priori* using a 2-D or 3-D field-solver based method, and then either a curve fitting or a table look up approach is used for characterization. During the extraction process, the patterns of the real structures are matched with the ones in the library and the per-unit-length capacitance numbers are obtained with interpolation and extrapolation [1.5] [1.6].

The simplest curve-fitting based model approximates the per-unit-length capacitance as

$$C = c_0 + c_1 w, \tag{1.3}$$

The process dependent constants c_0 and c_1, representing the edge and area capacitances, respectively, are calculated per layer. The form in (1.3) is widely used in wire sizing and for other optimization purposes.

For bends in the metal paths, vias, and other structures that require 3-D modeling, the capacitance values obtained from the extraction simply become capacitances from that node to ground.

For 2-D structures, the per-unit-length R and C values are used to model the total RC of the straight segment — often referred to as a uniform RC segment (URC). URCs are best described in the Laplace domain, making them somewhat incompatible with traditional transient-analysis algorithms. But due to the low pass nature of RC circuits, it can be shown that only a small number of lumped segments are required to accurately model a URC, and there is never a need for more than five lumped segments to model any URC for digital circuit applications [1.7]. Moreover, as interconnect paths pass over other metal layers, each metal crossing represents a discontinuity in the per-unit-length C, so there are few *uniform* RCs of sizeable length. Also, fewer than five lumps are adequate in most cases.

One conservative estimate for the number of lumped segments (N) required to model a URC, based on the maximum signal frequency of interest, is obtained by solving

$$f_{\max} \le \left| \frac{2N^2}{RC}\left(1 - \cos\frac{(2N-1)\pi}{2N}\right)\right| \tag{1.4}$$

for N [1.7].

1.2.2 RLC Trees

In RC tree modeling of interconnects, all of the neighboring conductors are modeled as ground in addition to substrate. This is a reasonable assumption as long as crosstalk effects are negligible. This model implies that the high frequency components of the current flowing in the interconnect return through the capacitors. But it does not say anything about the return path for dc and low frequency components of the current.

This is the main problem in on-chip inductance modeling because the inductance is defined only for closed loops, and it is very difficult to determine the return paths without a detailed analysis. Of course, as we demonstrate in the next subsection, an accurate analysis is possible via a partial inductance approach. However, this is very impractical if we want only a quick performance estimation of the interconnect.

In contrast, we show that the moment-based analysis of RLC trees is extremely easy. Thus for a quick estimation of inductive effects such as delay, ringing, and overshoot, RLC tree modeling of interconnects can be very useful. With some assumptions on the return paths [1.8], tree modeling is possible via the loop inductance concept. In reality, the current can return anywhere from the power/ground network of the chip. Thus, we can, for example, pick the closest power rail or stripe as the return path. Alternatively, we can assume (grounded) shields around the net that is being analyzed. Once a return path is identified, then it is straightforward to calculate the loop inductance from the partial self inductances of the signal and return conductors and the partial mutual inductance between them. Under the uniform current distribution assumption, which is the case if we neglect the skin and promixity effects, there exist closed form expressions for partial self and mutual inductances for certain types of geometries [1.9].

1.2.3 RLC Circuit Models

For the utmost accuracy of the interconnect parameter values, we can always start with Maxwell's equations and obtain a full-wave characterization by using either an integral equation or finite differencing method. Except for very few applications, however, this is not a desirable approach for interconnect analysis for two reasons: a-) the prohibitive runtime, b-) the difficulty in the modeling of the interaction with the nonlinear drivers and loads.

Circuit modeling of passive electromagnetic systems has been an active research area since the work of [1.10] which can be considered as the first attempt to relate Maxwell's equations to circuit models. For a review of electrical modeling of interconnects and packages, see [1.11]. Here, we will give a brief summary of the PEEC method, a popular approach to obtain circuit models from electromagnetic formulations.

As stated above, integral equation based methods are one of the two commonly employed techniques for electromagnetic modeling. An interesting property of the integral equation statement of an electromagnetic problem is that the integral equation can be interpreted in terms of the capacitive and inductive interactions between the

elemental currents and charges in the discretized structure. Thus, by introducing circuit elements it is possible to construct a complete distributed equivalent circuit description of the discrete electromagnetic problem. The resulting circuits are called *partial element equivalent circuits* (PEEC).

We use the notation (L_p, P, R, τ) PEEC to refer to the rigorous, full-wave PEEC model where L_p, P, and R represent the elements for inductance, potential (capacitance), and resistance, and τ indicates that retardation is included. However, depending on the application and the required accuracy, it may be possible to eliminate circuit elements from the full-wave PEEC model to obtain simpler models.

Typically, the on-chip and package structures are electrically small, i.e., the length and the overall physical size of such a structure is only a small fraction of the minimum wavelength in the bandwidth of interest. In such cases, the retardation is expected to have a minor effect. Hence, the full-wave modeling may not be necessary. Thus we can use the simple (L_p, P, R)PEEC models.

The capacitances in the nonretarded PEEC model are represented with the coefficients of potential. With a matrix inversion, we can convert the coefficients of potential P to capacitances C and the model will be a (L_p, C, R)PEEC model. Many of the techniques that are presented in this book are directly applicable to these types of circuits. For the sake of simplicity, they are referred to as RLC circuits.

For on-chip applications, it is often the case that the inductive effects can be neglected. For such cases, the model is referred to as a (C, R)PEEC model. In this book we refer to them as RC circuits.

We also note the RL modeling which is especially suitable for packaging structures. For the case where capacitance elements can be neglected, specialized solvers can be used for the solution of the resulting reduced (L_p, R) PEEC circuit [1.12].

A. Capacitance Extraction

The PEEC modeling starts with the coefficients of potential, and after a matrix inversion the capacitances are obtained for RC and RLC modeling.

More precisely, the conductor surfaces are divided into n small panels, and it is assumed that on each panel, the charge is uniformly distributed. We then solve a matrix equation of the form

$$\mathbf{Pq} = \mathbf{v} \tag{1.5}$$

where $\mathbf{P}$ is the $n \times n$ dense potential coefficient matrix, $\mathbf{q}$ is the vector of panel charges, and $\mathbf{v}$ is the vector of known panel potentials.

Direct inversion methods, such as Gaussian elimination, require $O(n^3)$ operations to solve (1.5). Clearly, this approach becomes computationally intractable for the sizes of the problems that typically appear in IC interconnect modeling. Instead, iterative algorithms such as GMRES are usually applied to solve the problem in (1.5). In this case, the dominant cost is the $O(n^2)$ operations required to compute $\mathbf{Pq}^{(k)}$ at each iteration, where $\mathbf{q}^{(k)}$ is the charge vector at the kth iteration. Thus, even the iterative techniques are very costly for typical interconnect problems. In addition, in terms of memory, both direct and iterative methods require $O(n^2)$ storage.

The main cost in iterative methods is the computation of the product $\mathbf{Pq}^{(k)}$ which is equivalent to computing n potentials from n charges. One approach to accelerate the matrix-vector products is to use potential approximations based on multipole expansions [1.13]. The basic concept is as follows. Consider n_1 potentials due to n_2 panels. The cost of the direct evaluation is $n_1 \cdot n_2$ operations. Now assume that n_2 panels are close to each other and n_1 potential evaluation points are all far away from the charges. With multipole approximations, the n_2 charges are replaced with a single charge equal to their sum. The number of operations to compute the n_1 potentials given this simplification is then $n_1 + n_2$, that is, n_2 operations to compute the sum of charges, and n_1 operations to calculate the potentials at the evaluation points. In addition, the matrix $\mathbf{P}$ is never explicitly built, hence a significant memory reduction is obtained.

In summary, with multipole expansions the cost per iteration can be reduced to $O(n)$ operations. A 3-D capacitance extraction program based on this approach, FASTCAP [1.14], has become very popular and has initiated an extensive amount of research in the area of the development of fast integral equation solution techniques.

We also note a stochastic approach for IC capacitance extraction. A floating random walk based method [1.15] has proven to be very effective for 3-D capacitance extraction.

1.3 Interconnect Analysis Via Moments

Once we have extracted values for the interconnect resistances, capacitances (and inductances), we can analyze the delay, rise-time, noise, etc., by various analysis and simulation algorithms. However, due to the nature and size of interconnect circuit models, one efficient solution approach is in terms of moments. In this subsection we will briefly outline the definition of moments, and explain how and why they are efficiently calculated in a linear circuit -- whether it is a simple RC tree or a large, strongly coupled RCL interconnect model.

Consider the simple RC ladder circuit shown in Figure 1.6. We can express the transfer function of this circuit as

$$H(s) = \frac{V_{out}(s)}{V_{in}(s)} = \frac{a_0 + a_1 s + a_2 s^2 + \ldots + a_n s^n}{1 + b_1 s + b_2 s^2 + \ldots + b_m s^m} \tag{1.6}$$

where $m > n$. Expanding (1.6) about $s = 0$ we can rewrite the transfer function as a series in powers of s:

$$H(s) = m_0 + m_1 s + m_2 s^2 + m_3 s^3 + \ldots \tag{1.7}$$

where the coefficients, m_j's, are known as circuit *moments*. Let $h(t)$ be the impulse response at the output. From the time-frequency domain relationship

$$H(s) = \int_0^\infty h(t) e^{-st} dt \tag{1.8}$$

it can be shown that

$$m_q = \frac{(-1)^q}{q!} \int_0^\infty t^q h(t) dt \,. \tag{1.9}$$

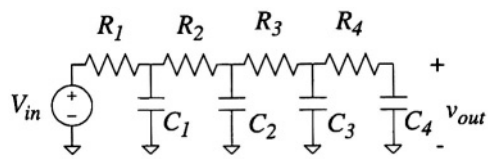


FIGURE 1.6 Simple RC ladder circuit

That is, the circuit moments, m_j's, are related to the probability moments of the impulse response $h(t)$ by the $(-1)^q/q!$ term.

In Chapter 5, we show that moments in linear circuits are generated very efficiently. For example, to find the moments of the circuit given in Figure 1.6, we recursively solve the dc equivalent circuit shown in Figure 1.7. In the case of m_0's the current sources are set to zero and the input source is left intact.

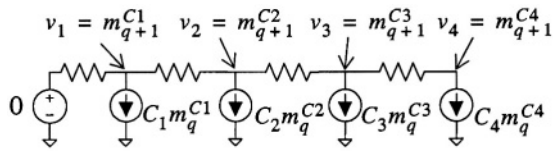


FIGURE 1.7 dc equivalent circuit to solve for moments.

Note that this procedure for replacing capacitors by current sources to calculate moments holds for all circuit topologies. A more complete explanation of the recursive procedure for calculating moments of general lumped, linear RLC circuits is described in terms of state variables in Chapter 4. Efficient calculation of moments for RLC tree-like topologies [1.16] and general circuits is described in Chapter 5.

1.4 Interconnect Metrics

Moments are extremely useful for analyzing RLC interconnect circuits via moment matching (Chapter 4) [1.17], but moments themselves are also useful as interconnect metrics. In particular, the first moment of the impulse response, the Elmore delay, is by far the most popular delay metric for RC interconnect trees. Similarly, for RLC trees, the first few moments can be used as metrics for delay and signal integrity control (Chapter 3).

1.4.1 Elmore Delay — First Moment

The Elmore delay [1.18], or the first moment of the impulse response, is a popular metric for RC trees since it is perhaps the most accurate delay metric that is a simple algebraic function of the R's and C's. Penfield and Rubenstein introduced this metric and the ease with which it is calculated for *RC* tree type problems [1.19]. Two $O(N)$ traversals of the tree, where N is the number of nodes in the tree, yield the Elmore delay.

The basis of Elmore's approximation lies in the observation that the impulse responses of circuits with monotonic step responses are nonnegative functions. Therefore, the impulse responses can be viewed as probability density functions. For example, the step and impulse responses of a simple RC tree circuit are shown in Figure 1.8. Since the step response is the integral of the impulse response $h(t)$, the 50% point delay of the monotonic step response is equal to the median point of the impulse response: $\int_0^{\tau} h(t)dt = 0.5$. Referring to Figure 1.8, Elmore proposed to approximate the median, τ, by the mean, or the first moment of the $h(t)$ distribution:

$$T_D = -m_1 = \int_0^{\infty} t\, h(t)dt \tag{1.10}$$

where, $\int_0^{\infty} h(t)dt = 1$. Clearly, the accuracy of the Elmore delay will be affected by the spread (variance) and skew (asymmetry) of the impulse distributions.

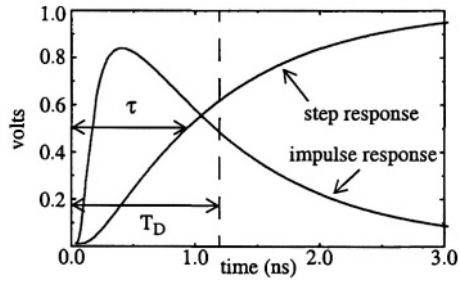


FIGURE 1.8 The unit step and the unit impulse (scaled) responses of a simple RC tree.

The skew, or the third central moment, μ_3, from distribution theory, measures the asymmetry of a distribution function. The second central moment, μ_2, represents the variance or spread. Their relation to circuit moment can be shown to be

$$\mu_2 = 2m_2 - m_1^2 \qquad \text{and} \qquad \mu_3 = -6m_3 + 6m_1m_2 - 2m_1^3 \tag{1.11}$$

In the next chapter, using central moments we prove that the Elmore delay is an absolute upper bound on the 50% delay of an RC tree response, even for finite rise-time input signals [1.20]. We further show that μ_2 and μ_3 are always positive for RC circuit responses, but their values vary with the shape of the impulse distribution. Only when two response nodes have similar shapes, hence μ_2 and μ_3, can we reliably use the first moment of the impulse response to approximate the delays with good relative accuracy.

1.4.2 Higher-Order Moment Metrics

Higher order delay approximations are needed to capture the DSM RC and RCL interconnect delays with reasonable relative accuracy. Higher order waveshape effects are also required to properly capture some of the gate behavior in terms of the effective capacitance loading. Some attempts have been made to formulate explicit solutions of second order (two time constant) models, which would seem to be the obvious compromise between an Elmore approximation and a complete reduced-order model via moment matching.

To apply such a two-pole model, however, requires a moment matching formulation that characterizes the poles in a provably stable manner. The first stable two-pole model was proposed in [1.21], but while it was stable, it could still produce complex pole pairs for RC circuits. In addition, this two-pole model required nonlinear iterations to solve for the delay, making it impractical for use during physical design.

In Chapter 3, we will provide some two-pole models which can be evaluated by approximate closed-form means or table-lookup. These approaches can be extended beyond second order, but it becomes increasingly difficult to evaluate the delays with the same efficiency that is possible with second order.

Another approach is to extend the delay metrics via the distribution interpretation proposed by Elmore. One approach is to fit RC tree/mesh impulse responses to time-shifted gamma functions via moment matching. Similarly, we can approximate homogenous parts of step responses with gamma distribution [1.22]. Importantly, these methods will be shown to be provably stable for RC trees and meshes based on their central moment properties. Because of the guaranteed stability, the delays for any percentage point of the waveform can be stored and evaluated via table lookup using the parameters of gamma that are a function of the first three moments.

1.4.3 RLC Metrics

Recently, metrics for RLC interconnects have been proposed based on the first three central moments [1.23]. Three moments are used to detect underdamping (ringing), specify proper termination, and estimate delay provided that the line is properly terminated. As lower resistance on-chip interconnects are developed, central moments have become important for detection and control of inductance effects.

1.4.4 Gate Delay Considerations

The percentage of the delay due to the RC interconnect has increased substantially relative to the gate delay for DSM technologies. As the metal resistance increases, the interconnect delay portion increases, and the gate delay component decreases due to resistance shielding effects. Roughly speaking, if the resistive component of the RC load is comparable or larger than the gate output impedance, the gate does not "see" all of the capacitance loading since the metal resistance "shields" some capacitance. This is most easily explained in terms of the circuit in Figure 1.9.

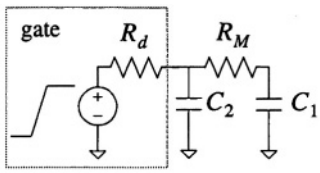


FIGURE 1.9 Gate driving RC load. Note that synthesized resistance, R_M is a function of the entire RC tree, and not determined simply by summing the metal resistances.

Referring to the figure, the driving point admittance of any RC load can be modeled by a reduced order circuit, in this case a pi-model, via moment matching [1.24]. If $R_d \gg R_M$, then the gate delay is accurately captured in terms of the total capacitance. However, as the metal resistance increases, and eventually, $R_d < R_M$, the gate delay will decrease, since the metal resistance will tend to shield some of the load capacitance.

Most gate level models are incompatible with RC interconnect loads, but the shielding effect should be captured to some degree of accuracy, at least for sensitivity analysis purposes. In order to preserve the simplicity and efficiency of these empirical gate models for complex RC loads, one can map the complex load to an *effective*

capacitance [1.25]. In Chapter 8, we show how we can derive approximate Thevenin equivalent models based on the Ceff concept which are suitable for use with physical design metrics [1.26].

1.5 Moment Matching and Model Order Reduction

If more moments are required for an accurate approximation, moment matching or other order reduction schemes can be used to generate reduced-order dominant pole/zero approximations for the interconnect transfer, admittance, and impedance functions. Later in this book, we cover such methods in detail. In the following we provide a brief summary.

A. Asymptotic Waveform Evaluation

Asymptotic Waveform Evaluation (AWE) [1.17] uses $2q$ moments to generate a q pole transfer function approximation, where q is much less than the order of the circuit.

If we expand the transfer function in (1.6) into its partial fractions,

$$H(s) = \frac{1 + a_1 s + a_2 s^2 + \ldots + a_n s^n}{1 + b_1 s + b_2 s^2 + \ldots + b_m s^m} = \sum_{j=1}^{m} \frac{k_j}{s - p_j} \tag{1.12}$$

we observe that for the case of distinct poles the time domain impulse response is

$$h(t) = \sum_{j=1}^{m} k_j e^{p_j t} \tag{1.13}$$

where the p_j's are the poles and the k_j's are the corresponding residues.

We can uniquely specify the poles and residues by forcing the first $2q$ moments of (1.13) to match the first $2q$ moments from (1.7) [1.17] [1.27]. Although this is recognized to be a form of Padé approximation, which is prone to producing unstable mod-

els of stable systems [1.28], we will discuss various algorithms which generate stable low-order models with excellent reliability.

B. Krylov-subspace techniques

When a large set of dominant poles are required, however, AWE may become ineffective due to inherent ill-conditioning nature of moment generation. In these cases, we can use the Krylov-subspace based order reduction methods [1.29] [1.30]. Instead of moments, these methods use Krylov vectors, which contain the same information as moments, but are better numerically conditioned. The generation of Krylov vectors is similar to that of moments. Except for some specific cases, however, the stability still remains a problem.

C. Passive Macromodeling

All of the efficient moment-based models for interconnect analysis are for *linear* circuits. The overall behavior and performance of a signal on the interconnect path is strongly dependent upon the nonlinear drivers and loads too.

One straightforward way of combining moment-based interconnect models and nonlinear components (e.g. transistors) is to characterize the linear interconnect portion of the circuit by a reduced order multiport (refer to Figure 1.10). For example, we can approximate the *Y* parameters in terms of the dominant poles and zeros. We then combine the reduced order interconnect models and the nonlinear devices in a circuit simulation environment. An important issue for such a simulation is the passivity. We can force the reduced order models to be stable, however, for a stable simulation the reduced order blocks have to be passive as well.

In Chapter 6, we present a guaranteed passive macromodeling algorithm, PRIMA [1.31]. For a brief overview of PRIMA, consider the modified nodal analysis description of a multi-input multi-output dynamic linear circuit

$$\begin{aligned} \mathbf{G}\mathbf{x} + \mathbf{C}\frac{d\mathbf{x}}{ct} &= \mathbf{B}\mathbf{u}_{\text{in}} \\ \mathbf{u}_{\text{out}} &= \mathbf{L}^T\mathbf{x} \end{aligned} \tag{1.14}$$

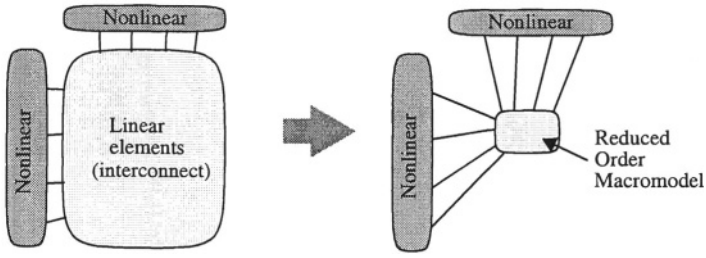


FIGURE 1.10 Model order reduction of multiport interconnect circuits.

where $\mathbf{u}_{in}$ and $\mathbf{u}_{out}$ are the vectors of inputs and outputs, respectively. PRIMA first computes a projection matrix $\mathbf{V}_q$, a block of Krylov vectors, and then applies projection to find

$$\begin{aligned} \mathbf{G}_q\mathbf{x}_q + \mathbf{C}_q\frac{d\mathbf{x}_q}{ct} &= \mathbf{B}_q\mathbf{u}_{in} \\ \mathbf{u}_{out} &= \mathbf{L}_q\mathbf{x}_q \end{aligned} \tag{1.15}$$

where the reduced order matrices are obtained as $\mathbf{G}_q = \mathbf{V}_q^T\mathbf{G}\mathbf{V}_q$, $\mathbf{C}_q = \mathbf{V}_q^T\mathbf{C}\mathbf{V}_q$, $\mathbf{B}_q = \mathbf{V}_q^T\mathbf{B}$, and $\mathbf{L}_q = \mathbf{V}_q^T\mathbf{L}$

Once a reduced order model is constructed, there are basically two approaches to interface them with SPICE-like circuit simulation environments. One option is to convert (1.15) into a state-space description and synthesize an equivalent circuit. The other approach is to express the reduced order system in the form of a pole-residue representation. We can then employ recursive convolution methods to obtain an equivalent discrete time-domain model for the reduced system during nonlinear circuit simulation [1.32]. The details of these approaches are described in Chapter 7.

1.6 Summary

As we have outlined in this chapter, this book covers a variety of interconnect analysis techniques with different efficiency-accuracy trade-offs. These techniques range from very simple delay metrics that can be used during the synthesis stage of IC design; to higher order delay and signal integrity metrics suitable for physical design; and conclude with accurate analysis methods that can be utilized in the final verification stages of chip design.

We begin with the Elmore delay -- the simplest of delay metrics -- in Chapter 2.

References

[1.1] H. B. Bakoglu, *Circuits, Interconnections, and Packaging for VLSI.* Reading, MA: Addison-Wesley Publishing Company, 1990.

[1.2] R. Ho, K. W. Mai, and M. A. Horowitz, "The future of wires," *Proc. IEEE*, vol. 89, no. 4, pp. 490-504, Apr. 2001.

[1.3] *International Technology Roadmap for Semiconductors*: Semiconductor Industry Association, 2001.

[1.4] T. Sakurai and K. Tamaru, "Simple formulas for two- and three-dimensional capacitances," *IEEE Trans. Electron Devices*, vol. ED-30, no. 2, pp. 183-185, Feb. 1983.

[1.5] U. Choudry and A. Sangiovanni-Vincentelli, "Automatic generation of analytical models for interconnect capacitances," *IEEE Trans. Computer Aided Design*, vol. 14, no. 4, pp. 470-480, Apr. 1995.

[1.6] N. Arora, K. V. Raol, R. Schumann, and L. M. Richardson, "Modeling and extraction of interconnect capacitances for multilayer VLSI circuits," *IEEE Trans. Computer Aided Design*, vol. 15, pp. 58-67, Jan. 1996.

[1.7] N. Gopal, D. P. Neikirk, and L. T. Pillage, "Evaluating RC-interconnect using moment-matching approximations," in *Tech. Dig. 1991 IEEE/ACM Int. Conf. on Computer-Aided Design*, Nov. 1991.

[1.8] Y.-C. Lu, M. Celik, T. Young, and L. Pileggi, "Min/max on-chip inductance models and delay metrics," in *Proc. 38th Design Automation Conference*, pp. 341-346, June 2001.

[1.9] F. Grover, *Inductance Calculations*. New York: Dover Publications, 1946.

[1.10] G. Kron, "Equivalent circuit for the field equations of Maxwell," *Proc. IRE*, vol. 32, no.289,1944.

[1.11] A. E. Ruehli and A. C. Cangellaris, "Progress in the methodologies for the electrical modeling of interconnects and electronic packages," *Proc. IEEE*, vol. 89, no. 5, pp. 740-771, May 2001.

[1.12] M. Kamon, M. J. Tsuk, and J. White, "Fasthenry: A multiple-accelerated 3-d inductance extraction program", *IEEE Trans. Microwave Theory and Techniques*, vol. 42, no. 9, Sept. 1994.

[1.13] L. Greengard and V. Rokhlin, "A fast algorithm for particle simulations," *J. Comp. Phys.*, vol. 73, pp. 325-348,1987.

[1.14] K. Nabors and J. White, "Fastcap: A multipole accelerated 3-D capacitance extraction program," *IEEE Trans. on Computer Aided Design*, vol. 10, no. 11, pp. 1447-1459, Nov. 1991.

[1.15] Y. L. LeCoz and R. B. Iverson, "A stochastic algorithm for high-speed capacitance extraction in integrated circuits," *Solid State Electronics*, vol. 35, pp. 1005-1012, 1992.

[1.16] C. L. Ratzlaff and L. T. Pillage, "RICE: Rapid interconnect circuit evaluation using AWE," *IEEE Transactions on Computer-Aided Design*, pp. 763-776, Jun. 1994.

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

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

[1.19] J. Rubenstein, P. Penfield, Jr., and M. A. Horowitz, "Signal delay in RC tree networks," *IEEE Trans. Computer-Aided Design*, vol. CAD-2, pp. 202-211, July 1983.

[1.20] R. Gupta, B. Tutuianu, and L. Pileggi, "The Elmore delay as a bound for RC trees generalized input signals," *IEEE Trans. Computer-Aided Design*, vol. 16, no. 1, Jan. 1997.

[1.21] M. A. Horowitz, *Timing Models for MOS Circuits*. Ph.D. thesis, Stanford University, January 1984.

[1.22] T. Lin, E. Acar, and L. T. Pileggi, "h-gamma: an RC delay metric based on a gamma distribution approximation of the homogeneous response," in *Tech. Dig. 1998 ACM/IEEE Int. Conf. on Computer-Aided Design*, pp. 19-24, Nov. 1998.

[1.23] R. Gupta, B. Krauter, and L. T. Pileggi, "Transmission line synthesis via constrained multivariable optimization," *IEEE Trans. on Computer-Aided Design*, vol. 16, no. 1, pp. 6-19, Jan. 1997.

[1.24] P. R. O'Brien and T. L. Savarino, "Modeling the driving-point characteristic of resistive interconnect for accurate delay estimation," in *Tech. Dig. IEEE/ACM Int'l Conference on Computer-Aided Design*, Nov., 1989.

[1.25] J. Qian, S. Pullela, and L. T. Pillage, "Modeling the effective capacitance for the RC interconnect of CMOS gate," *IEEE Trans. Computer-Aided Design*, vol. 12, no. 12, pp. 1526-1535, December 1994.

[1.26] F. Dartu, N. Menezes, J. Qian, and L.T. Pillage, "A gate-delay model for high-speed CMOS circuits," in *Proc. 31st ACM/IEEE Design Automation Conference*, June 1994.

[1.27] X. Huang, V. Raghavan, and R. A. Rohrer, "AWEsim: A program for the Efficient Analysis of Linear(ized) Circuits," in *Tech. Dig. IEEE/ACM Int'l Conference on Computer-Aided Design*, Nov. 1990.

[1.28] X. Huang, *Padé Approximation of Linear(ized) Circuit Responses*, PhD thesis, Dept. of Electrical Engineering, Carnegie Mellon University, Nov. 1990.

[1.29] P. Feldmann and R. W. Freund, "Efficient linear circuit analysis by Padé approximations via the Lanczos process," *IEEE Trans. on Computer-Aided Design of CAS*, vol. 14, no. 5, pp. 639-649, May 1995.

[1.30] L. M. Silveira, M. Kamon, and J. White, "Efficient reduced-order modeling of frequency-dependent coupling inductances associated with 3-D interconnect structures," in *Proc. 32nd IEEE/ACM Design Automation Conf*, pp. 376-380, San Francisco, Jun. 1995.

[1.31] A. Odabasioglu, M. Celik, and L. Pileggi, "PRIMA: Passive reduced-order interconnect macromodeling algorithm," *IEEE Trans. on Computer-Aided Design of CAS*, vol. 18, no. 8, pp. 645-654, Aug. 1998.

[1.32] V. Raghavan, E. Bracken, and R. A. Rohrer, "AWESpice: A general tool for the accurate and efficient simulation of interconnect problems," in *Proc. Design Automation Conference*, June 1992.

CHAPTER 2

The Elmore Delay

The simplest form of performance evaluation for IC interconnect is in terms of the delay metrics that are available for RC tree circuits. There has been a vast body of work on delay prediction for such circuits, and in this chapter we analyze in detail the Elmore delay --- the metric on which most explicit RC tree approximations are based.

The chapter begins with some necessary linear circuit terminology that enables us to introduce the Elmore delay as the first moment of the RC circuit impulse response. From there the connections made to higher-order circuit response moments and other circuit properties facilitate a more complete understanding of the Elmore delay limitations, and provide links to other measures of performance, such as phase delay.

2.1 Delay of a Transfer Function

Most interconnect structures are characterized by inputs that drive them and the resulting voltage at the outputs of interest. Such structures are often represented by transfer and immittance functions.

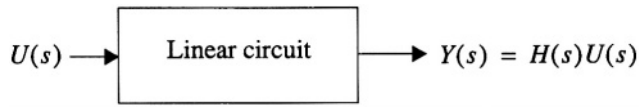


FIGURE 2.1 A single-input single-output circuit.

2.1.1 Circuit Transfer and Immittance Functions

Transfer functions are defined for linear circuits with zero initial conditions. Assume a single-input single-output linear circuit as shown in Figure 2.1. The transfer function is defined as the ratio of the output to the input:

$$H(s) = \frac{Y(s)}{U(s)} \tag{2.1}$$

An important subset of the transfer functions are the *immittance functions,* which are defined at the driving points and have two types. If the input is a current source and the output is the voltage measured across the source, the function is said to be *input impedance.* If the input is a voltage source and the output is the current through the source, the function is said to be *input admittance*. Often, $Z_{in}(s)$ and $Y_{in}(s)$ are used for input impedance and admittance functions, respectively. Throughout this book, however, we use $H(s)$ to denote all types of transfer and immittance functions, unless otherwise stated.

A transfer function is often expressed as a ratio of two polynomials in s :

$$H(s) = \frac{b_0 + b_1 s + \ldots + b_m s^m}{1 + a_1 s + \ldots + a_n s^n} \tag{2.2}$$

where the coefficients are real numbers. By factoring the polynomials in (2.2), an alternative representation of the transfer function is obtained:

$$H(s) = K\frac{(s - z_1) \cdot (s - z_2) \ldots (s - z_m)}{(s - p_1) \cdot (s - p_2) \ldots (s - p_n)} \tag{2.3}$$

where K is the multiplicative constant, z_i's are the *zeros,* and p_i's are the *poles* of the transfer function. The poles and zeros may be complex numbers appearing as conjugate pairs. Except for the constant K, a transfer function can be fully defined by a plot of poles and zeros in the complex plane such as shown in Figure 2.2.

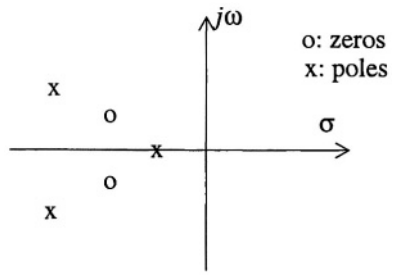


FIGURE 2.2 Example of a pole-zero plot in the complex plane.

Although pole-zero representation is very useful in the analysis and design of analog circuits, pole-residue form is preferable for time-domain interconnect analysis. Once the poles are obtained, the *residues,* k_i's, are calculated using partial fraction decomposition. Assuming distinct poles we can then express the transfer function as

$$H(s) = d + \sum_{i=1}^{n} \frac{k_i}{s - p_i} \tag{2.4}$$

where d is the direct coupling term.

2.1.2 Time-Domain Response and Delay

In linear circuits, the time- and s-domain responses are related to each other by Laplace transformations. Assume that $h(t)$ is the corresponding time-domain response for $H(s)$. Then we have

$$H(s) = \int_0^\infty e^{-st} h(t) dt \tag{2.5}$$

If the transfer function $H(s)$ is the complex frequency domain output when the input value is unity, $h(t)$ is the time-domain response for an impulse input and is known as

the *impulse response.* If a pole-residue representation such as (2.4) is available for the transfer function, the impulse response is given by the closed form expression

$$h(t) = d\delta(t) + \sum_{i=1}^{n} k_i e^{p_i t} u(t) \tag{2.6}$$

where $\delta(t)$ and $u(t)$ are the unit impulse and unit step functions, respectively. Once the impulse response poles and residues are known, the responses for step and ramp inputs can be expressed similarly.

For the case of a unit step input, from (2.1) the output response in the Laplace domain becomes

$$Y(s) = H(s) \cdot \frac{1}{s} \tag{2.7}$$

Using the pole-residue representation for the transfer function we obtain

$$Y(s) = \frac{d}{s} + \sum_{i=1}^{n} \frac{k_i}{(s - p_i)s} \tag{2.8}$$

With a partial fraction decomposition the above equation becomes

$$Y(s) = \frac{d}{s} + \sum_{i=1}^{n} \frac{k_i}{p_i}\left(\frac{1}{s - p_i} - \frac{1}{s}\right) \tag{2.9}$$

It follows that the step response can be written as

$$y(t) = \left(d + \sum_{i=1}^{n} \frac{k_i}{p_i}(e^{p_i t} - 1)\right) u(t) . \tag{2.10}$$

Similarly, for a unit ramp input denoted by $r(t)$, the s-domain output response is given by

$$Y(s) = H(s) \cdot \frac{1}{s^2} = \frac{d}{s^2} + \sum_{i=1}^{n} \frac{k_i}{p_i^2}\left(\frac{1}{s-p_i} - \frac{1}{s} - \frac{p_i}{s^2}\right), \tag{2.11}$$

and the ramp response becomes

$$y(t) = \left(dr(t) + \sum_{i=1}^{n} \frac{k_i}{p_i^2}(e^{p_i t} - 1 - p_i r(t))\right) u(t) \tag{2.12}$$

This approach of expressing time-domain responses using symbolic inverse Laplace transformations can be easily extended to any piecewise linear input waveform. Since any piecewise linear function can be written as a set of time-shifted ramp functions, the output waveform can be found by applying superposition and using (2.12).

One input waveform type, the saturated ramp as shown in Figure 2.3, is of particular interest because it is widely used for input waveform modeling in delay calculation in digital circuits. Such a waveform can be written as

$$v_i(t) = \frac{1}{t_r}(r(t) - r(t - t_r)) \tag{2.13}$$

Skipping the details, the expression for the output waveform is obtained as

$$v_o(t) = \frac{1}{t_r}(y(t) - y(t - t_r)) \tag{2.14}$$

where $y(t)$ is the ramp response given in (2.12).

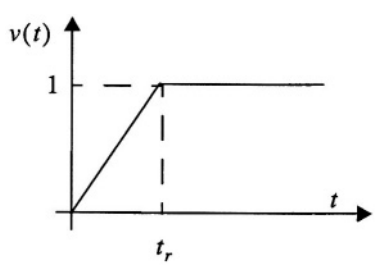


FIGURE 2.3 A saturated ramp waveform with a rise time t_r.

The propagation delay in a step response or a saturated ramp response can be obtained by solving (2.10) or (2.14), respectively. For example, assuming a monotonic response, an α percent delay for a step input signal is found by calculating τ in

$$d + \sum_{i=1}^{n} \frac{k_i}{p_i}(e^{p_i \tau} - 1) = \alpha \tag{2.15}$$

assuming that the steady-state response is also unity.

There are two major problems associated with this approach of delay calculation, and one of the goals of this book is to provide useful solutions for the delay problem. The first problem is obtaining the poles and residues. It is almost impossible to calculate all of the poles and residues for practical size interconnect circuits. A large portion of this book is dedicated to describing algorithms that capture only the dominant poles that are required to accurately approximate the time-domain responses. For this purpose, we will describe efficient order reduction methods such as asymptotic waveform evaluation and other Padé approximation based techniques.

But even with a small subset of dominant poles, solving for delay in (2.15) can be quite time consuming, especially in the inner loops of optimization processes. For this reason, we will introduce simpler delay metrics that represent the delay in closed form.

2.1.3 Attenuation and Phase Delay

In the frequency domain, the ac response is obtained by evaluating $H(s)$ on the $j\omega$ axis ($\omega = 2\pi f$) and is represented as

$$H(j\omega) = A(\omega)e^{j\theta(\omega)} \tag{2.16}$$

where $A(\omega)$ and $\theta(\omega)$ are the amplitude and the phase functions, respectively. The *attenuation* is defined as

$$\alpha(\omega) = -20\log A(\omega) \tag{2.17}$$

in decibel units. The *phase delay* is defined as

$$\tau(\omega) = -\frac{\theta(\omega)}{\omega} \tag{2.18}$$

where $\theta(\omega)$ is the phase function of the ac response. In terms of poles and zeros

$$\tau(\omega) = \frac{1}{\omega}\left[\sum_{i=1}^{n} \text{atan}\frac{-\omega}{p_i} - \sum_{i=1}^{m} \text{atan}\frac{-\omega}{z_i}\right] \tag{2.19}$$

Phase delay is the steady-state delay at the output waveform when a sinusoidal waveform at the angular frequency ω is applied to the circuit. For example, consider a system whose transfer function is given in (2.16) and assume a sinusoidal waveform as input, $v_i(t) = \cos\omega_0 t$. The output waveform is then given as

$$v_o(t) = A(\omega_0)\cos(\omega_0 t + \theta(\omega_0)) \tag{2.20}$$

which can also be expressed in the form

$$v_o(t) = A(\omega_0)\cos(\omega_0(t - T_{\text{delay}})) \tag{2.21}$$

Clearly, the delay term is

$$T_{\text{delay}} = -\frac{\theta(\omega_0)}{\omega_0} \tag{2.22}$$

2.2 RC Interconnect Delay

Whether it is a time- or frequency-domain analysis, the most often analyzed on-chip interconnect circuit model is in the form of an RC tree. RC trees, such as the one shown in Figure 2.4, have been widely used for modeling the gate and interconnect circuits like the one shown in Figure 2.5. As explained in the previous chapter, an RC tree is an RC circuit with capacitors from all nodes to ground, no capacitors between nonground nodes, and no resistors connected to ground. For modeling simplicity, the nonlinear driver in Figure 2.5 is linearized as shown in Figure 2.4. Linearization of gate models, and gate-interconnect interaction in general, are explained in detail in Chapter 8. In this chapter, we focus on estimating the delays of linearized RC trees, such as the one shown in Figure 2.4.

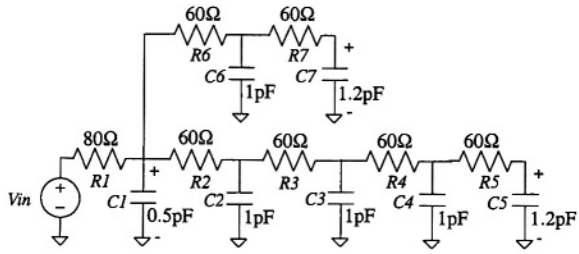


FIGURE 2.4 A simple RC tree circuit.

Once a gate and interconnect delay problem is translated into a linear RC tree problem, the delay can be estimated via a spectrum of approximation methods. The Elmore delay, or the first moment of the impulse response, is the most ubiquitous metric due to its simplicity. The main advantage of the Elmore delay is that it is a simple, closed form expression in terms of the RC tree parameter values. But we show in this chapter that the Elmore delay must be applied cautiously, especially for deep submicron technologies for which the first moment of the impulse response provides limited efficacy. For this reason, following this detailed coverage of the Elmore delay properties and limitations, more advanced delay metrics for RC and RLC trees are described in the chapter which follows.

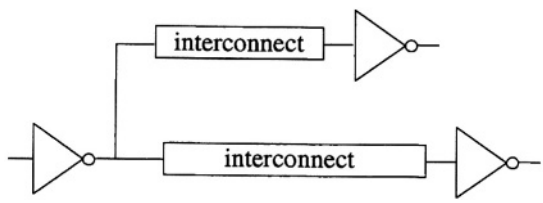


FIGURE 2.5 A CMOS inverter driving similar inverters through RC interconnect.

2.3 The Elmore Delay

The Elmore delay, introduced in 1948 for estimating the delay of amplifier circuits [2.1], was recognized by Penfield and Rubinstein as a useful metric for RC trees due to its step response properties [2.2]. For example, the step response for the node voltage at capacitor C_5 of the RC tree in Figure 2.4 is shown in Figure 2.6. Also shown in Figure 2.6 is the unit impulse response, $h(t)$, at the same node.

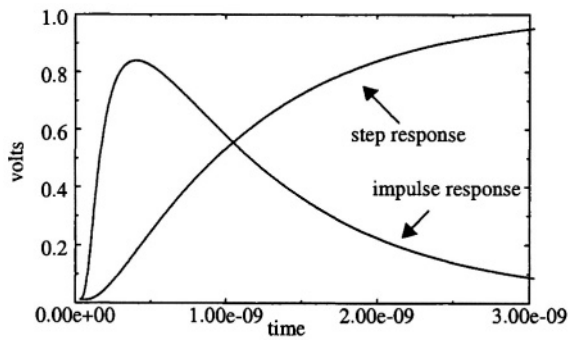


FIGURE 2.6 The unit step and the unit impulse response (scaled by 1e+09) for the voltage across C_5 in Figure 2.4.

Since the step response is the integral of the impulse response, the 50% point delay of the monotonic step response is the time τ at which the impulse response area has been half consumed:

$$\int_0^{\tau} h(t)dt = 0.5 \tag{2.23}$$

Referring to Figure 2.7, Elmore proposed to approximate τ, the delay of a monotonic step response, by the mean of the corresponding nonnegative impulse response function, $h(t)$. Treating the nonnegative impulse response in Figure 2.7 as a probability density function (p.d.f.), the mean is defined by the first moment of the impulse response. Elmore's unit step response delay approximation, T_D, is therefore

$$T_D = \int_0^{\infty} t\, h(t)dt \tag{2.24}$$

when the area underneath $h(t)$ equals unity

$$\int_0^\infty h(t)dt = 1 \tag{2.25}$$

This approximation appears valid for the symmetrical function in Figure 2.7, where the mean is equal to the median, however it is somewhat erroneous for the realistic impulse response in Figure 2.6, which is skewed asymmetrically. Later we use this skew to show that the mean, T_D, is an upper bound for the delay, τ, in RC trees.

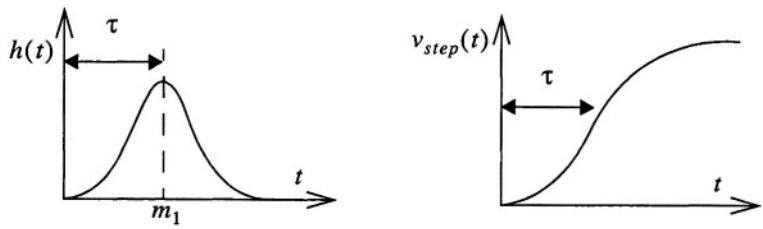


FIGURE 2.7 Elmore's approximation.

2.3.1 Calculating the Elmore Delay

The Elmore delay is a convenient metric for RC trees because it can be calculated very easily for this particular circuit topology. Efficient path tracing algorithms for calculating the Elmore delay and higher order moments for RLC trees are covered extensively later in this book, so they are not discussed in detail here. In summary, one can calculate the Elmore delays from two $O(N)$ traversals of the tree, where N is the number of nodes in the tree. The Elmore value for the output at node i is given by

$$T_{D_i} = \sum_{k=1}^{N} R_{ki}C_k \tag{2.26}$$

where R_{ki} is the resistance of the portion of the (unique) path between the input and node i that is common with the (unique) path between the input and node k, and C_k

is the capacitance at node *k*. For example for the circuit shown in Figure 2.4, the Elmore delay at node C_7 is given by

$$T_{D_7} = R_1C_1 + R_1C_2 + R_1C_3 + R_1C_4 + R_1C_5 + (R_1 + R_6)C_6 + (R_1 + R_6 + R_7)C_7 \quad (2.27)$$

The Elmore delay values at nodes C_1, C_5 and C_7 for the circuit in Figure 2.4 are compared with the exact delay in columns 2 and 3 of Table 2.1. In the same table some other delay metrics are also included, which are explained later in this chapter.

TABLE 2.1. Delay bounds for circuit in Figure 2.4.

(1)	(2)	(3)	(4)	(5)	(6)	(7)
Node	Actual delay	Elmore delay, T_D	Lower bound, $T_D - \sigma$	Single pole approx., $T_D \cdot \ln(2)$	PRH upper bound, t_{max}	PRH lower bound, t_{min}
C1	0.196 ns	0.55 ns	0 ns	0.383 ns	0.55 ns	0 ns
C5	0.919 ns	1.2 ns	0.2 ns	0.83 ns	1.32 ns	0.51 ns
C7	0.45 ns	0.75 ns	0 ns	0.524 ns	1.02 ns	0.054 ns

2.3.2 Dominant Pole Metric

In addition to the Elmore's classical definition of delay in terms of the first moment of the impulse response, we can also use the Elmore term to generate a dominant pole approximation. We demonstrate this relationship between the dominant pole and the first moment of the impulse response as follows.

Let $H(s)$ be a transfer function in an RC tree. Since the dc gain is unity in RC trees, $H(s)$ can be written in the normalized form

$$H(s) = \frac{1 + b_1 s + \ldots + b_m s^m}{1 + a_1 s + \ldots + a_n s^n} \quad (2.28)$$

Using (2.5) it can be shown that the first order moment is equivalent to

$$\int_0^\infty t\, h(t)dt = -\left.\frac{dH(s)}{ds}\right|_{s=0} \quad (2.29)$$

It follows that the Elmore delay can be expressed as

$$T_D = a_1 - b_1 \tag{2.30}$$

To understand the connection between the first moment and the dominant pole, we factor the numerator and the denominator of (2.28) and show that terms a_1 and b_1 are the sum of the reciprocal poles (circuit time constants) and the sum of the reciprocal zeros, respectively,

$$a_1 = -\sum_{j=1}^{n} \frac{1}{p_j} \qquad b_1 = -\sum_{j=1}^{m} \frac{1}{z_j} \tag{2.31}$$

If there are no low frequency zeros, the numerator coefficients, including b_1, are small and

$$T_D \cong a_1 \tag{2.32}$$

If one of the time constants (or poles) is dominant, that is,

$$\frac{1}{p_d} \gg \frac{1}{p_j}, \qquad j = 1, 2, \ldots, m, \quad j \neq d \tag{2.33}$$

then

$$T_D \cong -\frac{1}{p_d} \tag{2.34}$$

This dominant time constant approximation is then used to fit a single pole approximation. The corresponding step response becomes

$$v(t) = (1 - e^{p_d t})u(t) \tag{2.35}$$

Solving (2.35) for the 50% point delay effectively scales the Elmore delay by a constant factor

$$t_{\text{delay}} = \ln(2) T_D \approx 0.7 \cdot T_D \tag{2.36}$$

We should point out that this dominant time constant delay prediction can be either pessimistic or optimistic at two different nodes in the same RC tree. For example, column (5) of Table 2.1 shows the values of $\ln(2) \cdot T_D$ at nodes C_1, C_5 and C_7 for the circuit in Figure 2.4. Notice that, when compared with the actual delay values in column (1), the response at C_5 is optimistically predicted by $\ln(2) \cdot T_D$ while that at C_1 is pessimistically predicted. One way to explain this is by the excessive skew in the $h(t)$ distribution for C_1, which is shown with the step response for this node in Figure 2.8, as compared with the skew for the response at C_5 (shown in Figure 2.6). It can be expected that using ln (2) · Mean to approximate the *median* will be vastly different for these two distributions.

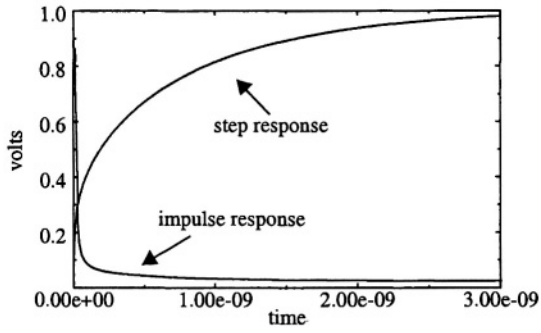


FIGURE 2.8 The unit step and the (scaled by 4e+09) unit impulse response for the voltage across C_1 in Figure 2.4

It is difficult to know when a single pole dominates the low-frequency behavior of a circuit. For this reason, Rubinstein and Penfield established bounds for the step response delay in RC trees.

2.3.3 Penfield and Rubinstein's Bounds

Penfield and Rubinstein were the first to use the Elmore delay to analyze RC trees [2.2]. Before higher-order moment matching techniques were available, delay bounds were the only means of estimating the accuracy of the RC tree delay approximation.

The Penfield and Rubinstein bounds require the definition of two new time constants,

$$T_P = \sum_k R_{kk} C_k$$
$$T_{R_i} = \left(\sum_k R_{ki}^2 C_k \right) / R_{ii} \tag{2.37}$$

in addition to the Elmore delay,

$$T_{D_i} = \sum_k R_{ki} C_k \tag{2.38}$$

where, as defined previously, R_{ki} is the resistance of the portion of the (unique) path between the input and node i that is common with the (unique) path between the input and node k, and C_k is the capacitance at node k.

Note that a circuit has one value of T_p, but each node has separate T_{R_i} and T_{D_i} values. From the definitions of R_{ki} and R_{ii} it can be easily shown that

$$T_{R_i} \le T_{D_i} \le T_p \tag{2.39}$$

Using these time constants, the step response lower bound in (2.40) and upper bound in (2.41) are obtained. For the details of the derivation, readers are referred to [2.2].

$$v_i(t) \ge \begin{cases} 0, & 0 \le t \le T_{D_i} - T_{R_i} \\ 1 - \dfrac{T_{D_i}}{t + T_{R_i}}, & T_{D_i} - T_{R_i} \le t \le T_p - T_{R_i} \\ 1 - \dfrac{T_{D_i}}{T_p} e^{(T_p - T_{R_i} - t)/T_p} & T_p - T_{R_i} \le t \le \infty \end{cases} \tag{2.40}$$

$$v_i(t) \le \begin{cases} 1 - \dfrac{T_{D_i} - t}{T_p} & 0 \le t \le T_{D_i} - T_{R_i} \\ 1 - \dfrac{T_{R_i}}{T_p} e^{(T_D - T_{R_i} - t)/T_{R_i}} & T_{D_i} - T_{R_i} \le t \le \infty \end{cases} \tag{2.41}$$

Lower and upper bounds on an α percent delay are found from the step response bounds:

$$t_{\min}(\alpha) = \begin{cases} 0, & 0 \le \alpha \le 1 - \dfrac{T_{D_i}}{T_P} \\ T_{D_i} - T_P(1-\alpha), & 1 - \dfrac{T_{D_i}}{T_P} \le \alpha \le 1 - \dfrac{T_{R_i}}{T_P} \\ T_{D_i} - T_{R_i} + T_{R_i} \ln\left[\dfrac{T_{R_i}}{T_P(1-\alpha)}\right], & 1 - \dfrac{T_{R_i}}{T_P} \le \alpha < 1 \end{cases} \tag{2.42}$$

$$t_{\max}(\alpha) = \begin{cases} \dfrac{T_{D_i}}{1-\alpha} - T_{R_i}, & 0 \le \alpha \le 1 - \dfrac{T_{D_i}}{T_P} \\ T_P - T_{R_i} + T_P \ln\left[\dfrac{T_{D_i}}{T_P(1-\alpha)}\right], & 1 - \dfrac{T_{R_i}}{T_P} \le \alpha < 1 \end{cases} \tag{2.43}$$

Calculating these bounds requires calculating two additional terms in addition to the Elmore delay. All of these terms, however, are obtained with $O(N)$ complexity using a path tracing algorithm. The values of $t_{\max}$ and $t_{\min}$ at the 50% point for our example in Figure 2.4 are given in columns (6) and (7) of Table 2.1. Note that $t_{\max} > T_D$ at the loads, C_5 and C_7, and $t_{max} = T_D$ at the driving point, C_1. Also note the values of $t_{\min}$ as a lower bound on the delay.

2.4 Moments

The first moment of the impulse response defines the Elmore delay, but it is through higher order moments that we begin to understand the limitations of this metric. To understand higher order moments, consider a transfer function in the Laplace domain

$$H(s) = \frac{b_0 + b_1 s + b_2 s^2 + \ldots + b_m s^m}{1 + a_1 s + a_2 s^2 + \ldots + a_n s^n} \tag{2.44}$$

Expanding (2.44) about $s = 0$ we can express the transfer function as an infinite series in powers of s

$$H(s) = m_0 + m_1 s + m_2 s^2 + m_3 s^3 + \ldots \tag{2.45}$$

where

$$m_q = \frac{1}{q!} \frac{d^q H(s)}{ds^q}\bigg|_{s=0} \tag{2.46}$$

The time-frequency domain relationship follows from the Laplace transform of $h(t)$

$$H(s) = \int_0^\infty h(t) e^{-st} dt . \tag{2.47}$$

Expanding e^{-st} about $s = 0$ in (2.47) yields

$$\begin{aligned} H(s) &= \int_0^\infty h(t)\left[1 - st + \frac{1}{2}s^2t^2 - \frac{1}{6}s^3t^3 + \ldots\right] dt \\ &= \sum_{k=0}^{\infty} \frac{(-1)^k}{k!} s^k \int_0^\infty t^k h(t) dt \end{aligned} \tag{2.48}$$

It follows from (2.48) that the qth coefficient of the impulse response $h(t)$ is

$$m_q = \frac{(-1)^q}{q!} \int_0^\infty t^q h(t) dt \tag{2.49}$$

In probability distribution theory the qth moment of a function $h(t)$ is defined as [2.3]

$$q\text{-th probability moment: } \int_0^\infty t^q h(t) dt , \tag{2.50}$$

Thus the coefficients, m_q's, are related to the probability moments by the $(-1)^q/q!$ term. For ease of notation, however, in this book we refer to the series coefficients in (2.45) and (2.49) as moments. When necessary, we distinguish them by explicitly stating as circuit moments and probability moments.

2.4.1 Calculating Moments

Moments are very useful design metrics because of the ease with which they are calculated, particularly for RC or even RLC trees. Efficient calculation methods, including a path tracing algorithm for RLC trees, will be explained later in this book. But, we now demonstrate the concept with the simple RC tree circuit shown in Figure 2.9.

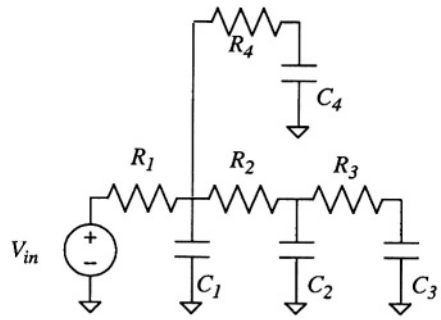


FIGURE 2.9 A simple RC tree circuit example.

The impulse response of this circuit in the Laplace domain can be analyzed in terms of the circuit in Figure 2.10, where capacitors have been replaced by their complex admittances. Let us assume that each of the capacitor voltages (which in this circuit also are the node voltages) is expressed in terms of an infinite series in powers of s, as shown in the figure. The superscripts for the m_j's in Figure 2.10 denote that all of the m_j's are different from one node to the next.

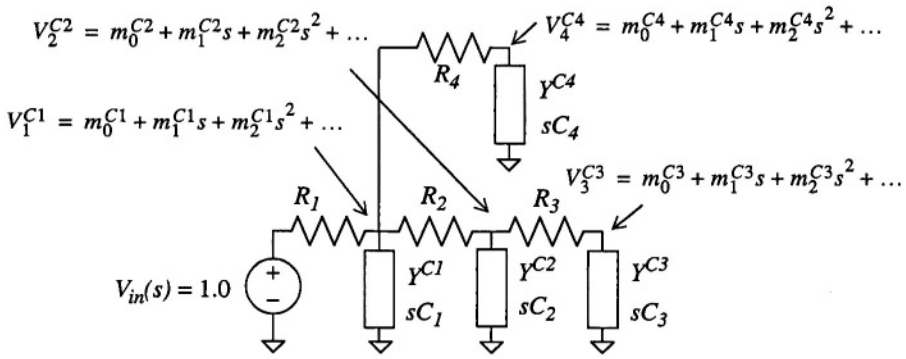
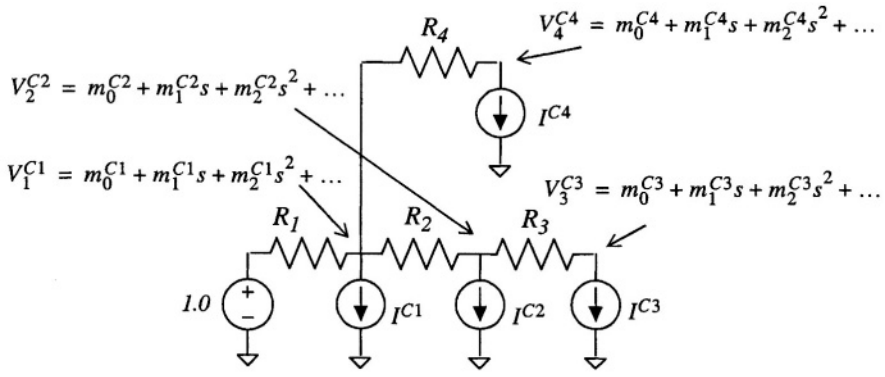


FIGURE 2.10 The s-domain representation of the RC circuit in Figure 2.9 in terms of complex admittances.

Expressing the capacitor voltages in this way and knowing the capacitor admittances, we can write similar expressions for the capacitor currents, as shown in Figure 2.11. Moreover, knowing the capacitor currents in terms of the capacitor voltages, we can replace the complex admittances by current sources, as shown in Figure 2.11. The m_j terms are the only unknowns in Figure 2.11.



$$I^{C1} = sC_1(m_0^{C1} + m_1^{C1}s + m_2^{C1}s^2 + \ldots) \qquad I^{C3} = sC_3(m_0^{C3} + m_1^{C3}s + m_2^{C3}s^2 + \ldots)$$

$$I^{C2} = sC_2(m_0^{C2} + m_1^{C2}s + m_2^{C2}s^2 + \ldots) \qquad I^{C4} = sC_4(m_0^{C4} + m_1^{C4}s + m_2^{C4}s^2 + \ldots)$$

FIGURE 2.11 A circuit equivalent to Figure 2.10 assuming the node voltages solutions of the form shown.

Referring to Figure 2.11, we can solve for the m_0's for all of the capacitor voltages by setting $s = 0$. Since there are no constant terms (s^0 terms) in the capacitor currents (they are open for $s = 0$), we set the current sources in Figure 2.11 to zero and solve for the m_0's using the dc equivalent circuit in Figure 2.11. For this RC circuit, the m_0's are all equal to 1.0. Note that this procedure for replacing capacitors by zero valued current sources to calculate the m_0's holds for all circuit topologies. As we explain later, when there are inductors in the circuit, they are replaced by zero valued voltage sources when calculating the m_0 terms for their current responses.

Referring back to Figure 2.11, we now solve for the s^1 coefficients, m_1's. It is the s-terms in the current source expressions which produce the m_1's in the voltage responses. Since we know the m_0's, the s-terms in the current source expressions are known. Therefore, we can evaluate the m_1's of the voltage responses by setting all of

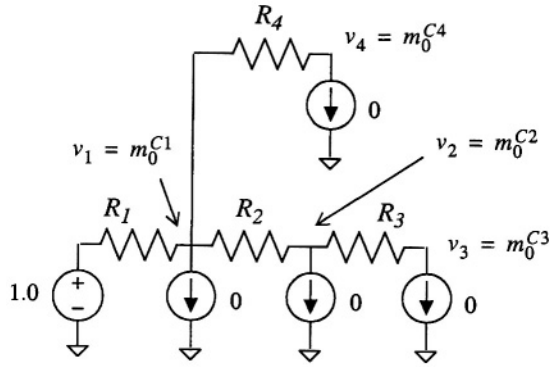


FIGURE 2.12 The dc equivalent circuit used to calculate the m_0's for all of the capacitor voltages.

the current sources equal to $C_j m_0^j$, and solving for the node voltages, which are the m_1^j's. The voltage input is a constant, so it does not affect the calculation of any of the terms other than the m_0's. All subsequent moments are calculated from the dc equivalent circuit shown in Figure 2.13 following the same recursion.

With the ability to calculate higher order moments, we can now consider delay approximations that include higher order terms.

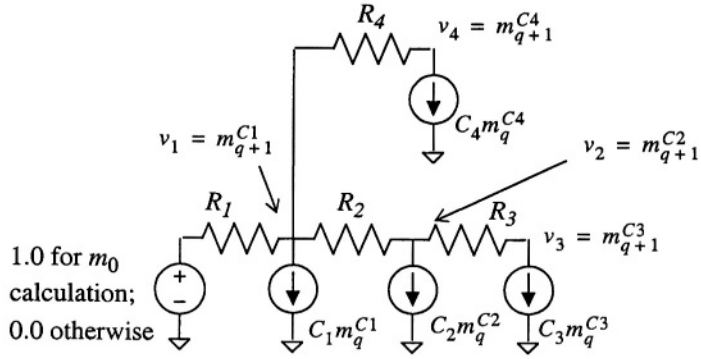


FIGURE 2.13 The dc equivalent circuit used to calculate the moments for the RC circuit in Figure 2.9.

2.5 Extending Elmore's Distribution Theory Analogy

The accuracy of the Elmore delay depends on the relative positioning of the mean and the median of the impulse response distribution. There are no error approximations for the Elmore delay, but we can measure its accuracy relative to the information from higher order moments. Keeping with Elmore's analogy of viewing an RC impulse response as a p.d.f., we can characterize the higher-order waveshape terms by central moments.

2.5.1 Central Moments

Similar to moments, central moments are distribution theory concepts. Following Elmore's distribution function analogy, we can use them to explain the properties of Elmore delay approximation. As we demonstrate in the next chapter, they are also useful measures for RLC circuit responses.

Consider the moment definition given in (2.49) again:

$$m_q = \frac{(-1)^q}{q!}\int_0^\infty t^q h(t)dt \tag{2.51}$$

The *mean* of the impulse response is given by [2.3]

$$\mu = \frac{\int_0^\infty th(t)dt}{\int_0^\infty h(t)dt} = \frac{-m_1}{m_0} \tag{2.52}$$

When the dc gain of the system is unity, $m_0 = 1$, the mean becomes $\mu = -m_1$.

Central moments of the impulse response are the moments about the *mean* and are defined as [2.3]

$$\mu_k = \int_0^\infty (t-\mu)^k h(t)dt \tag{2.53}$$

It is straightforward to show from (2.53) that the first few central moments can be expressed in terms of circuit moments as follows:

$$\mu_0 = m_0 \tag{2.54}$$

$$\mu_1 = 0 \tag{2.55}$$

$$\mu_2 = 2m_2 - \frac{m_1^2}{m_0} \tag{2.56}$$

$$\mu_3 = -6m_3 + \frac{6m_1m_2}{m_0} - 2\frac{m_1^3}{m_0^2} \tag{2.57}$$

Unlike the moments of the impulse response, the central moments have geometrical interpretations:

- μ_0 is the area under the curve. It is generally unity, or else a simple scaling factor is applied.
- μ_2 is the *variance* of the distribution which measures the spread or the *dispersion* of the curve from the center. A larger variance reflects a larger spread of the curve.
- μ_3 is a measure of the *skewness* of the distribution; for a unimodal function its sign determines if the mode (global maximum) is to the left or to the right of the expected value (mean). Its magnitude is a measure of the distance between the mode and the mean.

2.5.2 Second and Third Central Moments in RC Trees

The second and third central moments are always positive for RC tree impulse responses. The positiveness of the second order central moment is obvious from its definition

$$\mu_2 = \int_0^\infty (t-\mu)^2 h(t)dt \tag{2.58}$$

The impulse response, $h(t)$, at any node in an RC tree is always positive (see Appendix 2.A.4) and so is the integrand in (2.58). Hence the second central moment, μ_2, is always positive.

The proof of the positiveness of the third central moment in RC trees is given in Appendix 2.A.3.

2.6 The Elmore Delay as a Bound

Referring back to Figure 2.6, it is apparent that with such an asymmetrical distribution for the impulse response, the mean would not coincide with the median. In this section, we show that these asymmetric distributions have a "long tail" on the right side of the mode, which is roughly the maximum value point, and a "short tail" on the left side. Such distributions are said to have *positive* skew. We also prove that the impulse response for an RC tree is unimodal and then use these two properties to prove that

$$\text{Mode} \leq \text{Median} \leq \text{Mean} \tag{2.59}$$

Equation (2.59) states that the Elmore delay, or the mean of the impulse response, is truly an upper bound on the median, or the 50% point delay. We will show that this holds for any input that has a unimodal derivative and that the mean becomes a better approximation of the median as the rise-time of the input-signal increases. Further in the section, we also provide a lower bound on the 50% delay for an RC tree, but first a few definitions:

Definition 1: *The mode, M, of a distribution function is that value of the variate exhibited by the greatest number of members of the distribution [2.3]. If the distribution function f is continuous and differentiable, a unique mode exists only if f is unimodal and is the solution of*

$$f'(x) = \frac{d}{dx}f(x) = 0, \qquad f''(x) = \frac{d^2}{dx^2}f(x) < 0 \tag{2.60}$$

Definition 2: *The median, m, of a distribution function f is that value of the variate which divides the total frequency into two equal halves [2.3], i.e.*

$$\int_{-\infty}^{m} f(x)dx = \int_{m}^{\infty} f(x)dx = \frac{1}{2} \tag{2.61}$$

Definition 3: *The mean,* μ, *of a distribution function f is defined by*

$$\mu = \int_{-\infty}^{\infty} xf(x)dx \tag{2.62}$$

Definition 4: *A density function f (x) is called unimodal, if and only if, there exists at least one value* $x = x_m$ *such that f(x) is nondecreasing for* $x < x_m$ *and nonincreasing for* $x > x_m$ *[2.4].*

Definition 5: *The coefficient of skewness for a distribution function is given by* $\gamma = \mu_3/\sigma^3$, *where* $\sigma = \sqrt{\mu_2}$, *and* μ_2 *and* μ_3 *are the second and third central moments of the distribution function respectively [2.3].*

Lemma 1: *The impulse response h(t) at any node of an RC tree is a unimodal and positive function.*

The proof is given in Appendix 2.A.4.

Lemma 2: *For the impulse response h(t) at any node of an RC tree, the coefficient of skewness,* γ, *is always nonnegative.*

Proof: In Subsection 2.5.2, it was shown that in RC trees the second and third central moments are always positive. Thus, from Definition 5, for every node in an RC tree, the coefficient of skewness, $\gamma \geq 0$. QED

Theorem: *For the impulse response h(t) at any node in an RC tree,*

$$\text{Mode} \leq \text{Median} \leq \text{Mean} \tag{2.63}$$

Proof: For a unimodal "skewed" distribution function, the mean, median, and mode inequality states that these three quantities occur either in alphabetical order or reverse alphabetical order [2.5], i.e. either $\text{Mean} \leq \text{Median} \leq \text{Mode}$ or $\text{Mode} \leq \text{Median} \leq \text{Mean}$. From Lemma 1 and Lemma 2, each node in an RC tree has a unimodal distribution function for which $\gamma \geq 0$. We now prove by contradiction, that for an RC tree we have that $\text{Mode} \leq \text{Median} \leq \text{Mean}$.

For our contradiction argument, let $\text{Mean} \leq \text{Median} \leq \text{Mode}$ hold for any node, *k*, in an RC tree. In a symmetrical distribution, for which the coefficient of skewness, γ, is exactly zero, the mean, the median and the mode coincide [2.5], [2.6]. Thus a natural measure of skewness for an asymmetrical distribution is the deviation of the *mean* from the *median,* or the *mean* from the *mode.* Thus,

$$\text{Skew} = \frac{\text{Mean} - \text{Median}}{\sigma} \tag{2.64}$$

where $\sigma = \sqrt{\mu_2}$. Thus at node *k,* since $\text{Mean} \leq \text{Median} \leq \text{Mode}$ holds, skew is negative. But, from Lemma 2, we have that the coefficient of skewness, $\gamma \geq 0$. Thus, at node *k,* either Skew = 0 or we have a contradiction. In the former case, Mean = Median = Mode, i.e., the distribution is symmetric and the mean and median coincide. And in the latter case, $\text{Mode} \leq \text{Median} \leq \text{Mean}$. Since the choice of the node *k* is arbitrary, the proof is complete. QED

We should note at this point that the Elmore delay, T_D, or the mean, μ, of the impulse response approaches the 50% delay point at nodes further downstream from the source in an RC tree [2.7]. Thus, as one moves away from the source, μ is a better approximation of the net delay, as further discussed in Section 2.7.

A. A Lower Bound on Delay

Consider the following corollary whose proof is given in Appendix 2.A.5.

Corollary 1: *A lower bound on the 50% delay for an RC tree is given by*

$$\max \quad (\mu - \sigma, 0) \tag{2.65}$$

where μ *is the mean and* $\sigma = \sqrt{\mu_2}$.

Referring back to the example in Figure 2.4 and the delay bounds in Table 2.1, the $\mu - \sigma$ lower bound at C_1 equals $t_{\min}$, whereas at C_5 and C_7, $t_{\min}$ is a tighter lower bound than $\mu - \sigma$. However, the Elmore delay upper bound, μ, becomes a tighter upper bound at the leaf-nodes of an RC tree as is evident at C_5 and C_7 in Table 2.1.

B. Approximating the Output Signal Transition Time

Another measure of practical importance for RC circuits, other than the 50% delay point, is the rise-time, T_R, which may be defined as the time required for the response to increase from 10 to 90 percent of its final value. A good measure of the value of T_R for an output response is

$$T_R \propto \sigma = \sqrt{\mu_2} \tag{2.66}$$

where μ_2 is the second central moment of the output response. Elmore also proposes this value, which he terms the *radius of gyration,* as a rise-time measure for step-responses [2.1].

2.7 The Elmore Delay for General Input Signals

In the previous section, it was shown that the Elmore delay is an upper bound on the 50% step response delay. In addition, with one more moment the variance can be calculated to establish a lower bound on the 50% delay. However, when using the Elmore delay to estimate RC interconnect delays, the signal coming out of the digital gate is never a step voltage, and is generally modeled by a saturated ramp. For this reason, we extend this Elmore-based bound to consider non-zero input signal transition time, or more appropriately, the variance of the input signal's derivative.

A. The Elmore Delay Upper Bound

Consider a saturated ramp input voltage $v_i(t)$ with a transition time t_r as shown in Figure 2.14. It is easy to show that its derivative $v_i'(t)$ has the following properties:

$$\int_0^\infty v_i'(t)dt = 1$$
$$\mu_2[v_i'(t)] = \frac{t_r^2}{12} \qquad (2.67)$$
$$\mu_3[v_i'(t)] = 0$$

Now consider the following result for the general case of the input signals with non-negative $\mu_2[v_i'(t)]$ and $\mu_3[v_i'(t)]$ values.

Corollary 2: *For an RC circuit with a monotonically increasing, piecewise-smooth input* $v_i(t)$ *such that* $v_i'(t)$ *is a nonnegatively skewed unimodal function,* $\text{Mode} \le \text{Median} \le \text{Mean}$ *holds for the output response* $v_o(t)$ *at any node.*

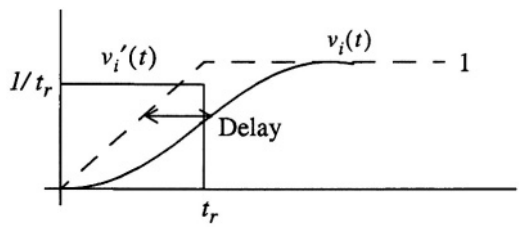


FIGURE 2.14 Input signal $v_i(t)$ with rise-time, t_r and its derivative, $v_i'(t)$.

Proof: The output response $v_o(t)$ at any node of an RC tree in response to an input $v_i(t)$ is given in the Laplace domain by

$$V_o(s) = H(s)V_i(s) \tag{2.68}$$

where $H(s)$ is the transfer function of the circuit at that node. Also, $v_i(t)$ is a piecewise-smooth function and hence piecewise differentiable (refer to Figure 2.14. Thus, assuming that $v_o(0) = 0$ and $v_i(0) = 0$, we obtain

$$\begin{aligned} \mathcal{L}\{v_o'(t)\} &= s\mathcal{L}\{v_o(t)\} - v_o(0) \\ &= s\mathcal{L}\{h(t)^* v_i(t)\} \\ &= \mathcal{L}\{h(t)\}\mathcal{L}\{v_i'(t)\} \end{aligned} \tag{2.69}$$

where, $\mathcal{L}(.)$ is the Laplace transform operator. Further, from Appendix 2.A.2, we have the property that the second and third central moments add under convolution. Thus,

$$\begin{aligned} \mu_2[v_o'(t)] &= \mu_2[h(t)] + \mu_2[v_i'(t)] \\ \mu_3[v_o'(t)] &= \mu_3[h(t)] + \mu_3[v_i'(t)] \end{aligned} \tag{2.70}$$

From Subsection 2.5.2, we know that $\mu_2[h(t)] \geq 0$ and $\mu_3[h(t)] \geq 0$. From hypothesis, we also have

$$\mu_2[v_i'(t)] \geq 0 \qquad \text{and} \qquad \mu_3[v_i'(t)] \geq 0 \tag{2.71}$$

From (2.70) and (2.71), therefore, $\mu_2[v_o'(t)] \geq 0$ and $\mu_3[v_o'(t)] \geq 0$. Thus, from Definition 5, $\gamma[v_o'(t)] \geq 0$, and Median $\leq$ Mean. QED

Corollary 3: *Assume a finite sized RC circuit with a monotonically increasing, piecewise-smooth input $v_i(t)$ such that $v_i'(t)$ is a symmetric function. Then, as the rise-time of the input signal, $t_r \to \infty$, the Elmore delay for the output response, $T_D \to$ 50% Delay, i.e.* Mean $\to$ Median.

Proof: The output response $v_o(t)$ at any node of an RC tree in response to an input $v_i(t)$ is given in the Laplace domain by

$$V_o(s) = H(s)V_i(s) \tag{2.72}$$

And from (2.69) and (2.70),

$$\begin{aligned}\mu_2[v_o'(t)] &= \mu_2[h(t)] + \mu_2[v_i'(t)] \\ \mu_3[v_o'(t)] &= \mu_3[h(t)] + \mu_3[v_i'(t)]\end{aligned} \tag{2.73}$$

From hypothesis, we have that $v_i'(t)$ is a symmetric function. Therefore, $\mu_3[v_i'(t)] = 0$. Also, since $\mu_2[v_i'(t)] \propto t_r$,

$$t_r \to \infty \quad \Rightarrow \quad \mu_2[v_i'(t)] \to \infty \tag{2.74}$$

Also from hypothesis, the circuit is finite sized, that is, $|\mu_3[h(t)]| < \infty$. Thus,

$$\gamma[v_o'(t)] = \frac{\mu_3[v_o'(t)]}{(\mu_2[v_o'(t)])^{3/2}} \to 0 \quad \text{as} \quad t_r \to \infty \tag{2.75}$$

Since $\gamma \propto \text{Mean} - \text{Median}$, $\gamma \to 0 \Rightarrow \text{Mean} \to \text{Median}$. Thus, as the rise-time of the input-signal increases without bound, the 50% delay for an RC tree approaches the Elmore delay, T_D. QED

It is noteworthy here that since $\mu_3[v_i'(t)] = 0$, i.e. $v_i'(t)$ is a symmetric function, its mean and median coincide. Further

$$\begin{aligned}\mu[v_o'(t)] - \mu[v_i'(t)] &= T_D \\ \Rightarrow \int_0^\infty t v_o'(t)dt - \int_0^\infty t v_i'(t)dt &= T_D\end{aligned} \tag{2.76}$$

Integrating by parts,

$$\begin{aligned}\Rightarrow \left\{\int_0^\infty [1 - v_o(t)]dt - t[1 - v_o(t)]\Big|_0^\infty\right\} - \left\{\int_0^\infty [1 - v_i(t)]dt - t[1 - v_i(t)]\Big|_0^\infty\right\} &= T_D \\ \Rightarrow \int_0^\infty [1 - v_o(t)]dt - \int_0^\infty [1 - v_i(t)]dt &= T_D\end{aligned} \tag{2.77}$$

where we have used the fact that $\lim_{t\to\infty} t[1 - v_o(t)] = 0$ and $\lim_{t\to\infty} t[1 - v_i(t)] = 0$ since both $v_i(t)$ and $v_o(t) \to 1V$ exponentially as $t \to \infty$ [2.8]. Thus, (2.77) says that the area between the input and the output response equals the Elmore Delay, T_D [2.9].

B. Ramp Follower Responses

In summary, as depicted in Figure 2.15, when the input signal transition time is much larger than the largest circuit time constant, the transient response is negligible, and the output follows the steady-state asymptote. This kind of response corresponds to a ramp follower response in which the Elmore delay becomes nearly exact. Note that in a typical IC circuit most of the nets behave like ramp followers. To demonstrate this, consider the histogram in Figure 2.16 which displays the Elmore delay errors for 1200 (ramp) responses for RC trees taken from a 0.35 micron CMOS microprocessor design. Some of the errors are quite small, while a significant percentage of the errors are greater than 50%. On the other hand, the dominant pole metric, which scales the Elmore delay by 0.7, does not change the relative delay error problem, but simply shifts the delays as shown by the histogram plot in Figure 2.16. Notice that with this dominant pole approximation, a large population of delays are underestimated. These are the responses for which the unsealed Elmore delays were nearly exact.

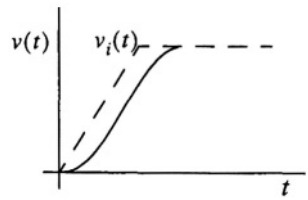


FIGURE 2.15 With a large risetime, the 50-50% delay approaches the Elmore delay.

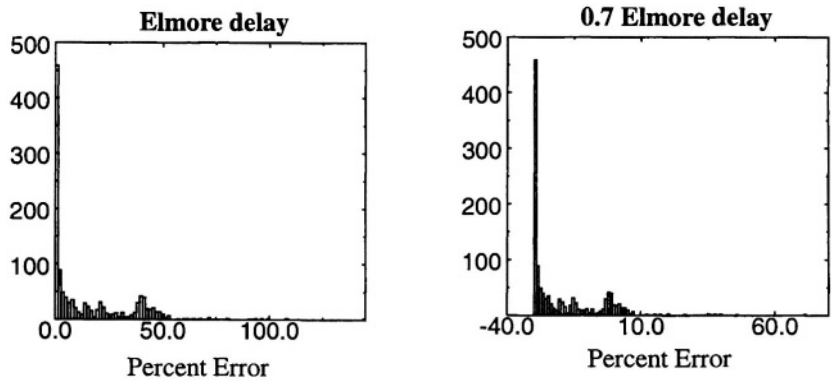


FIGURE 2.16 The Elmore and dominant time constant delay errors with respect to SPICE for an industrial IC with over 700 nets and 1200 load points.

C. Delay Curves

The estimation of the 50% delay by the Elmore delay as a function of the rise-time of the input signal (see Figure 2.14), as stated in Corollary 3, is shown in Figure 2.17 for our RC tree example circuit (in Figure 2.4). As the rise-time of the input signal increases, the delay asymptotically approaches the Elmore Delay value, T_D, as expected.

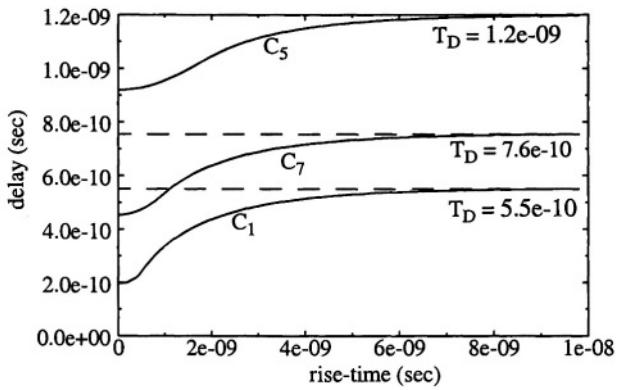


FIGURE 2.17 Delay curves show that as the rise time of the input signal increases, the delay approaches T_D.

It was observed earlier that as one moves away from the source, T_D (i.e. the mean, μ) is a better approximation of the net delay. The proof for Lemma 1 uses the additive property of the central moments under convolution. As proven in Subsection 2.5.2, for any node k, $\mu_2(h_k), \mu_3(h_k) \geq 0$. Furthermore, it can also be shown that as one moves further from the driving point, $\mu_2(h_{k,k+1})$ and $\mu_3(h_{k,k+1})$ form decreasing and hence convergent sequences [2.7]. Thus, as nodes farther away from the source are considered, the values of $\mu_2(h_k)$ and $\mu_3(h_k)$ start to converge and hence the skew, γ, converges. The fact that T_D is a better approximation of the net delay farther away from the driving point is illustrated here using a 25 node RC tree. For three nodes A, B and C, where A is near the driving point, B is in the middle of the tree and C is a leaf-node, the impulse responses are shown in Figure 2.18. The response at node C is less "asymmetric" than the response at node B, which shows that the impulse response approaches symmetry away from the driving point and the Elmore delay T_D becomes a tighter bound on the 50% delay point.

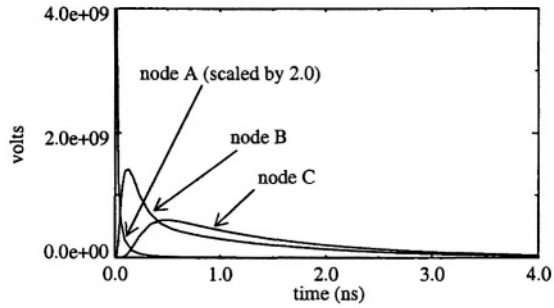


FIGURE 2.18 Impulse responses at nodes A (driving point), B (middle node) and C (leaf node).

Table 2.2 shows the relative errors $(\text{Delay} - T_D)/\text{Delay}$ for different input signal rise-times.

TABLE 2.2. Delays and relative error at nodes A, B, C along a signal path.

Node	Elmore delay	Rise-time = 1ns		Rise-time = 5ns		Rise-time =10ns	
		Delay	% Error	Delay	% Error	Delay	% Error
A	0.02 ns	0.01 ns	104%	18.0 ps	11.9%	19.0 ps	1.54%
B	1.13 ns	0.72 ns	54.7%	1.06 ns	6.5%	1.116 ns	0.86%
C	1.56 ns	1.2 ns	29.6%	1.48 ns	4.8%	1.547 ns	0.64%

2.8 The Elmore Delay as a Bound for Phase Delay In RC Trees

In addition to its relationship to properties of the RC impulse response, the Elmore delay can also be shown to be an upper bound on the phase delay of RC tree responses. To show this relation between the Elmore delay and the phase delay, we start with the properties of general RC circuit impedance functions and generalize them for transfer functions in RC circuits.

The poles and zeros of impedance functions of general RC circuits have the following properties [2.10]:

1. All poles and zeros are located on the negative real axis of the *s*-plane.
2. Poles and zeros are interlaced.
3. The lowest critical frequency is a pole which may be at the origin.

The second property is, in general, not true for general transfer functions. In Appendix 2.A.6, we show that the transfer functions in RC tree circuits have the following properties:

1. All poles and zeros are located on the negative real axis of the *s*-plane.
2. For each zero there is a corresponding pole which is located between that zero and the origin (poles and zeros need not to be interlaced).
3. The lowest critical frequency is a pole which may be at the origin.

From these properties it follows that an RC tree transfer function can be written in the form

$$H(s) = K\frac{(s-z_1)\cdot(s-z_2)\ldots(s-z_m)}{(s-p_1)\cdot(s-p_2)\ldots(s-p_n)} \tag{2.78}$$

where $n > m$, $p_1 > p_2 > \ldots > p_n$, $z_1 > z_2 > \ldots > z_m$, and $p_i > z_i, \quad i \le m$. Using (2.19) and the pole-zero notation in (2.78), the phase delay can be written as

$$\tau(\omega) = \frac{1}{\omega}\left[\sum_{i=1}^{m}\left(\operatorname{atan}\frac{-\omega}{p_i} - \operatorname{atan}\frac{-\omega}{z_i}\right) + \sum_{i=m+1}^{n}\operatorname{atan}\frac{-\omega}{p_i}\right] \tag{2.79}$$

Using the facts that $x > \operatorname{atan} x$ for $x > 0$ and $(x - \operatorname{atan} x) > (y - \operatorname{atan} y)$ for $x > y > 0$, we obtain

$$\tau(\omega) \le \left(\sum_{i=1}^{m}\frac{1}{z_i} - \sum_{i=1}^{n}\frac{1}{p_i}\right) \tag{2.80}$$

Previously we have shown that

$$T_D = \sum_{i=1}^{m} \frac{1}{z_i} - \sum_{i=1}^{n} \frac{1}{p_i} \tag{2.81}$$

From (2.80) and (2.81), it follows that

$$\tau(\omega) \leq T_D \tag{2.82}$$

It can be also shown that

$$T_D = \lim_{\omega \to 0} \tau(\omega) \tag{2.83}$$

Not surprisingly, this proof demonstrates that the phase delay approaches the Elmore delay as the signal frequency decreases. This is analogous to the manner in which the 50% delay for an RC tree asymptotically approaches the Elmore delay as the rise time of the input signal increases.

As an example, we consider an unbalanced clock tree from [2.11]. The phase delay at a response node is compared with the Elmore delay in Figure 2.19. Note that the Elmore delay is a tight upper bound up to a certain frequency point. Importantly, in the next chapter we show that this frequency point represents the maximum operating frequency for which the signal attenuation is likely to be acceptable. This demonstrates that the Elmore metric is an ineffective performance measure without a companion metric for the signal attenuation.

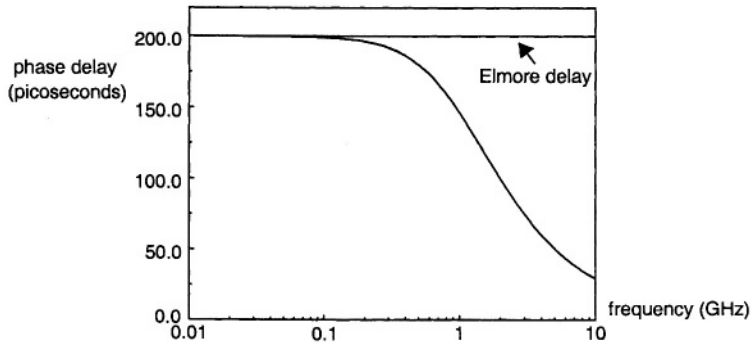


FIGURE 2.19 Comparison of the phase delay with the Elmore delay for an unbalanced clock tree. Note that beyond 10 GHz the phase delay keeps decreasing as the frequency increases as expected by the denominator term in (2.18).

2.9 Summary

The Elmore delay is an extremely popular timing performance metric which is used at all levels of electronic circuit design automation. It is provably an upper bound on the actual 50% delay of an RC tree step response and also holds for input signals with finite rise time. A lower bound on the actual delay can also be developed as a function of the Elmore delay and the second moment of the impulse response. Improved bounds may be theoretically possible with more moments, but the moment matching techniques that we describe in later chapters, such as AWE, are preferable when higher order moments are available. The utility of this bound is for understanding the accuracy and the limitations of the Elmore metric as we use it as a simple closed-form performance metric.

2.A Appendix

2.A.1 Driving Point Moments Have Alternating Signs

We prove that the signs of the moments of the driving point admittance functions in RC trees alternate.

Let $Y(s)$ be a driving point admittance function in an RC tree, and consider its representation in terms of poles and residues:

$$Y(s) = k_0 + \sum_{j=1}^{n} \frac{k_j}{s - p_j} \tag{2.84}$$

where n is the order of the circuit. Now consider a moment expansion for $Y(s)$:

$$Y(s) = y_0 + y_1 s + y_2 s^2 + \ldots \tag{2.85}$$

In terms of the poles and residues, the moments are given as

$$y_i = -\sum_{j=1}^{n} \frac{k_j}{p_j^{i+1}}, \qquad i > 0 \tag{2.86}$$

Using (2.86) and the fact that the poles and residues of admittance functions in RC trees are all real and negative [2.10], it can be easily shown that

$$\begin{aligned} y_i &< 0 \qquad \text{if } i = 2, 4, \ldots \\ y_i &> 0 \qquad \text{if } i = 1, 3, \ldots \end{aligned} \tag{2.87}$$

2.A.2 Central Moments Add Under Convolution

Consider two impulse responses, $h_1(t)$ and $h_2(t)$, and their convolution, $h(t)$,

$$h(t) = h_1(t) * h_2(t) \tag{2.88}$$

Assume that $h_1(t)$ and $h_2(t)$ are RLC tree impulse responses, i.e.,

$$m_0(h_1) = 1 \text{ and } m_0(h_2) = 1 \tag{2.89}$$

Then, the first few moments of $h(t)$ can be written in terms of the moments of $h_1(t)$ and $h_2(t)$ as follows:

$$\begin{aligned} m_0(h) &= 1 \\ m_1(h) &= m_1(h_2) + m_1(h_1) \\ m_2(h) &= m_2(h_2) + m_1(h_1)m_1(h_2) + m_2(h_1) \end{aligned} \tag{2.90}$$

From (2.56) and (2.90), the second central moment of $h(t)$ is expressed as

$$\begin{aligned} \mu_2(h) &= 2m_2(h_1) + 2m_2(h_2) - [m_1(h_1)]^2 - [m_1(h_2)]^2 \\ &= \mu_2(h_1) + \mu_2(h_2) \end{aligned} \tag{2.91}$$

From (2.57) and (2.90), the third central moment of $h(t)$ is expressed as

$$\begin{aligned} \mu_3(h) &= -6m_3(h_1) - 6m_3(h_2) + 6m_1(h_1)m_2(h_1) + 6m_1(h_2)m_2(h_2) \\ &\quad -2[m_1(h_1)]^3 - 2[m_1(h_2)]^3 \\ &= \mu_3(h_1) + \mu_3(h_2) \end{aligned} \tag{2.92}$$

2.A.3 The Positiveness of the Third Order Central Moments in RC Trees

We prove the positiveness of the third central moment by induction. Consider a general RC tree as shown in Figure 2.20. The remaining portion of the tree beyond node 1 is represented by the driving point admittance, $Y_1(s)$, at that node. The impulse response at node 1 is given by

$$V_1(s) = \frac{1}{1 + R_1 Y_1(s)} \tag{2.93}$$

where R_1 is the resistance between the source and the node 1. We next consider the moment expansion of the driving point admittance:

$$Y_1(s) = y_1 s + y_2 s^2 + y_3 s^3 + \ldots \tag{2.94}$$

Since there is no dc current path to ground, the zeroth order moment is zero. Inserting (2.94) in (2.93) and expanding $V_1(s)$ in terms of its moments

$$V_1(s) = \frac{1}{1 + R_1(y_1 s + y_2 s^2 + \ldots)} = 1 + m_1 s + m_2 s^2 + m_3 s^3 + \ldots \tag{2.95}$$

Multiplying the denominator with the right hand side expansion and equating the terms with the same powers of s, we calculate the first few moments in terms of admittance function moments:

$$\begin{aligned} m_1 &= -R_1 y_1 \\ m_2 &= R_1^2 y_1^2 - R_1 y_2 \\ m_3 &= R_1 y_1 (R_1 y_2 - R_1^2 y_1^2) + R_1^2 y_1 y_2 - R_1 y_3 \end{aligned} \tag{2.96}$$

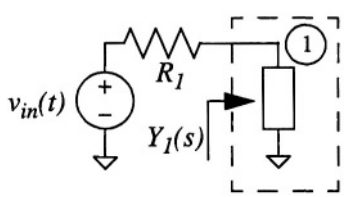


FIGURE 2.20 An RC tree and its driving point admittance at the first capacitor node.

The third central moment at node 1 is then given by

$$\begin{aligned}\mu_3(V_1) &= -6m_3 + 6m_1m_2 - 2m_1^3 \\ &= -6R_1^2y_1y_2 + 2R_1^3y_1^3 + 6R_1^2y_3\end{aligned} \tag{2.97}$$

In Appendix 2.A.1, it is shown that y_1 and y_3 are always positive and y_2 is always negative for RC tree driving point admittance functions. Therefore, it can be easily shown that $\mu_3(V_1) > 0$.

Next consider Figure 2.21 which shows node k and its "downstream" part of the RC tree. To complete the induction argument, assume that at node k, $\mu_3 \geq 0$ for $v_k(t)$. The transfer function at node $k + 1$ is given by

$$V_{k+1}(s) = V_k(s)H_{k,k+1}(s) \tag{2.98}$$

where the transfer function from node k to node k+1,

$$H_{k,k+1}(s) = \frac{1}{1 + R_kY_{k+1}(s)} \tag{2.99}$$

is exactly in the same form with (2.93). Therefore, if $h_{k,k+1}(t)$ is the impulse response which corresponds to $H_{k,k+1}(s)$, then we have $\mu_3(h_{k,k+1}) > 0$. In Appendix 2.A.2, it is shown that the third central moment add under convolution. Thus

$$\mu_3(V_{k+1}) = \mu_3(V_k) + \mu_3(H_{k,k+1}) > 0 \tag{2.100}$$

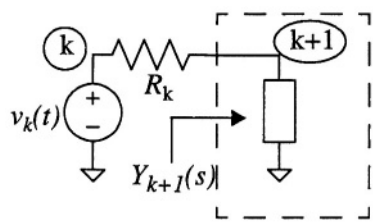


FIGURE 2.21 Admittance Y_{k+1} of an RC tree at an arbitrary node $k + 1$.

2.A.4 RC Tree Impulse Responses Are Unimodal and Positive Functions

We prove that the impulse response $h(t)$ at any node of an RC tree is a unimodal and positive function.

The proof is by induction. Consider a general RC circuit as shown in Figure 2.22. The remaining portion of the tree beyond node one is represented by the driving point admittance, $Y_1(s)$, at that node. Assuming that $v_i(t)$ is a unit impulse input, we will first prove that the voltage at node 1, $v_1(t)$, is unimodal and positive.

For a general RC circuit, the poles and residues of the driving point admittance, $Y_{\text{in}}(s)$ (in Figure 2.22) are real and negative [2.10]. Since $I_{\text{in}}(s) = V_i(s)Y_{\text{in}}(s) = Y_{\text{in}}(s)$, where $I_{\text{in}}(s)$ is the s-domain representation of the driving point current, $i_{\text{in}}(t)$, we can write $i_{\text{in}}(t)$ in terms of the poles and residues of $Y_{\text{in}}(s)$;

$$i_{\text{in}}(t) = \sum_i k_i e^{p_i t}, \quad k_i < 0,\ p_i < 0, \quad \text{for } t > 0 \tag{2.101}$$

Note that the above exponential form for the input current is valid only for $t > 0$. At $t = 0$, the current also contains an impulse component, which is skipped for the sake of simplicity since we are only interested when $t > 0$. From KCL, $v_1(t) = v_i(t) - i_{in}(t)R_1$. Since the voltage at any node is the impulse response at that node, the impulse response at node 1 is given by

$$\begin{aligned} h_1(t) &= -i_{\text{in}}(t)R_1, \quad t > 0 \\ &= -\left(\sum_i k_i e^{p_i t}\right)R_1, \quad t > 0 \end{aligned} \tag{2.102}$$

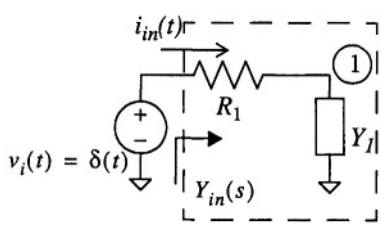


FIGURE 2.22 Input node of an RC tree with input admittance $Y_{\text{in}}(s)$.

where we used $v_i(t) = 0$ for $t > 0$. Since all k_i's and p_i's are real negative numbers, $h_1(t)$ is positive and monotonically decreasing. Therefore, from Definition 4, $h_1(t)$ is unimodal.

Now consider Figure 2.23 which shows node k and the RC tree "downstream" from node k. For the induction argument, we assume that $h_k(t) = v_k(t)$ is positive and unimodal, and then prove that the voltage, or the impulse response, at node $k + 1$,

$$v_{k+1}(t) = v_k(t) - R_k i_{k,k+1}(t) , \tag{2.103}$$

is positive and unimodal.

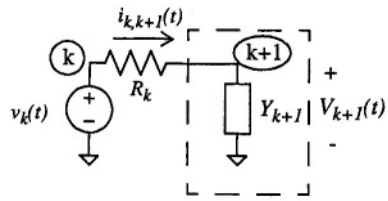


FIGURE 2.23 Admittance Y_{k+1} of an RC tree at an arbitrary node $k + 1$.

If $v_k(t)$ is an impulse input at node k, then $h_{k,k+1}(t)$ is the impulse response at node $k+1$ for the tree rooted at node k. This has the same form as in (2.102) and is unimodal. Thus, the impulse response at node $k + 1$ w.r.t. node 1 (the driving point), $h_{k+1}(t)$, is given by

$$h_{k+1}(t) = h_{k,k+1}(t) * h_k(t) \tag{2.104}$$

where * is the convolution operator. Since the convolution of two unimodal positive functions is also a unimodal positive function [2.4], $h_{k+1}(t)$ is positive and unimodal. Thus, $h(t)$ at any node of an RC tree is a positive unimodal function.

2.A.5 Proof of Corollary 1

We prove that a lower bound on the 50% delay for an RC tree is given by

$$\max \quad (\mu - \sigma, 0) \tag{2.105}$$

where μ is the mean and $\sigma = \sqrt{\mu_2}$.

Consider an impulse response $h(t)$, shown in Figure 2.24, with mean at $t = \mu$. We define another function $g(t)$ as

$$g(t) = \int_{-\infty}^{t} h(\zeta)d\zeta \tag{2.106}$$

With a simple change in the x coordinate such that $\tau = t - \mu$, we have $h(\tau)$ such that its mean is at $\tau = 0$ in the new coordinate system. Then, we use the following inequality from [2.3]:

$$g(\tau) \leq \frac{\sigma^2}{\sigma^2 + \tau^2}, \qquad \tau < 0 \tag{2.107}$$

For $\tau = -\sigma$, equations (2.106) and (2.107) show that

$$g(-\sigma) = \int_{-\infty}^{-\sigma} h(\zeta)d\zeta \leq \frac{\sigma^2}{\sigma^2 + (-\sigma)^2} = \frac{1}{2} \tag{2.108}$$

Equation (2.108) states that in the new coordinate system, $\tau = -\sigma$ is less than the median. Thus, in the original coordinate system, for $h(t)$ we have that $\mu - \sigma \leq Median$.

When $\mu \leq \sigma$, since the RC tree system is causal and relaxed for $t < 0$, we have $\text{Median} \geq 0$, and hence $\text{Median} \geq \max(\mu - \sigma, 0)$.

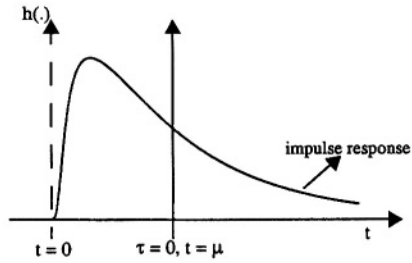


FIGURE 2.24 Impulse response $h(t)$ at an arbitrary node of an RC tree.

2.A.6 Properties of Poles and Zeros in RC Transfer Functions

Consider a general RC tree circuit, such as the one shown in Figure 2.25. We use the following notation to describe the circuit. Referring to Figure 2.25, let us assume that V_k is the output voltage of interest and let V_i $(0 < i < k)$ be the voltage of the ith node between the source and the load. We denote the capacitance between node i and ground by C_i and the resistance connected between nodes i-1 and i by R_i. From the Norton equivalent of the circuit, shown in Figure 2.25, the ac response at node 1 is found as

$$V_1(s) = \frac{Z_1(s)}{R_1} \tag{2.109}$$

where $V_{in}(s) = 1$ and $Z_1(s)$ is the input impedance seen from the current source of the second circuit in Figure 2.25 and satisfies the properties of impedance functions of RC circuits, which are listed in Section 2.8. Using the substitution theorem we obtain the equivalent circuit and its Norton equivalent in Figure 2.25. Now, the ac response at node 2 is given by

$$V_2(s) = \frac{V_1(s) \cdot Z_2(s)}{R_2} \tag{2.110}$$

Now consider the zeros of $V_2(s)$. Again referring to Figure 2.25, $V_2(s)$ becomes zero when the other branch that leaves node 1 becomes short circuit, e.g., the zeros of $V_2(s)$ are the poles of the driving point admittance function of that sub tree. This

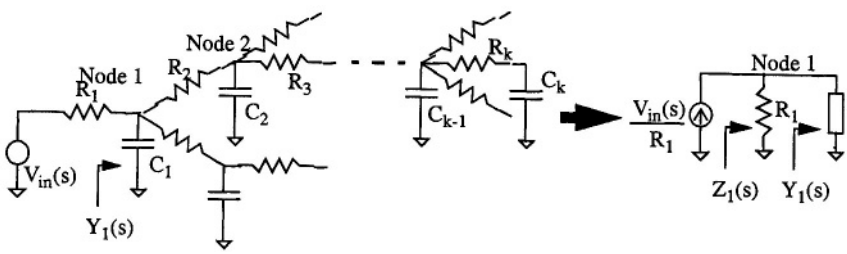


FIGURE 2.25 A general RC tree circuit and its Norton equivalent at Node 1.

implies that the zeros of the transfer functions in an RC tree circuits are on the negative real axis. From (2.110), the magnitude response at node 2 is found as

$$|V_2(j\omega)| = \frac{|V_1(j\omega)| \cdot |Z_2(j\omega)|}{R_2} \tag{2.111}$$

Since the poles and zeros of $Z_2(s)$ are interlaced, with the lowest critical frequency being a pole, its magnitude response, $|Z_2(j\omega)|$, is monotonically decreasing. Therefore

$$|V_2(j\omega)| < c \cdot |V_1(j\omega)| \tag{2.112}$$

where c is a constant. Note that $V_1(s)$ and $V_2(s)$ have exactly the same set of poles. Since the zeros of both of them are on the negative real axis, from (2.112) we conclude that some of the zeros of $V_2(s)$ have moved left.

Processing downstream in this way we observe that as we approach the output node the zeros either move left, some of them to infinity, or remain at their original positions. As a summary, the transfer function can be written in the form

$$V_k(s) = \frac{(s-z_1)\cdot(s-z_2)\ldots(s-z_m)}{(s-p_1)\cdot(s-p_2)\ldots(s-p_n)} \tag{2.113}$$

where $n > m$, $p_1 > p_2 > \ldots > p_n$, $z_1 > z_2 > \ldots > z_m$, and $p_i > z_i$, $\quad i \le m$

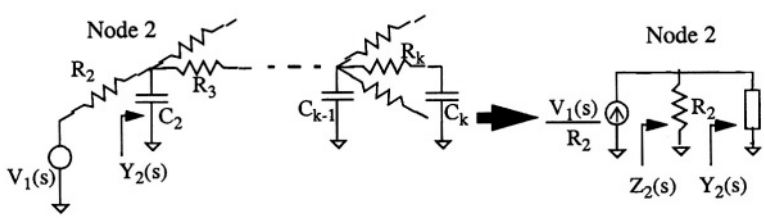


FIGURE 2.26 The equivalent circuit seen from Node 1 and its Norton equivalent at Node 2.

References

[2.1] W. C. Elmore, "The transient analysis of damped linear networks with particular regard to wideband amplifiers," *J. Applied Physics,* vol. 19(1), 1948.

[2.2] J. Rubinstein, P. Penfield, Jr., and M. A. Horowitz, "Signal delay in RC tree networks," *IEEE Trans. on Computer Aided Design,* vol. 2, pp. 202-211, 1983.

[2.3] H. Cramer, *Mathematical Methods of Statistics.* Princeton University Press, 1946.

[2.4] E. N. Protonotarios and O. Wing, "Theory of nonuniform RC lines. Part II: Analytic properties in the time domain," *IEEE Transactions on Circuit Theory,* March 1967

[2.5] M. G. Kendall and A. Stuart, *The Advanced Theory of Statistics. Vol. I: Distribution Theory.* Hafner Publishing Company, 1969.

[2.6] H. L. MacGillivray, "The mean, median, mode inequality and skewness for a class of densities," *Australian Journal of Statistics,* 23(2), 1981.

[2.7] R. Gupta, B. Tutuianu, and L. Pileggi, "The Elmore delay as a bound for RC trees with generalized input signals," *IEEE Trans. Computer-Aided Design,* vol. 16, no. 1, Jan. 1997.

[2.8] J. L. Wyatt, *Circuit Analysis, Simulation and Design,* Chapter on *Signal Propagation Delay in RC Models of Interconnect.* North-Holland, 1987.

[2.9] T. -M. Lin and C. A. Mead, "Signal delay in general RC networks," *IEEE Trans. on Computer Aided Design,* vol. CAD-3, pp. 331-349, 1984.

[2.10] M. E. Van Valkenburg, *Introduction to Modern Network Synthesis.* John Wiley & Sons Inc., 1960.

[2.11] M. Celik and L. Pileggi, "Metrics and bounds for phase delay and signal attenuation in RC(L) clock trees," *IEEE Trans. on Computer Aided Design,* vol. CAD-18, Mar. 1999.

CHAPTER 3

Higher-Order RC(L) Delay Metrics

Although it provides a simple and explicit delay metric for interconnect circuits that can be modeled by RC trees, the Elmore delay can sometimes exhibit poor accuracy. As the first moment of the impulse response, it cannot include the effects of resistive shielding, which is evident for long resistive lines; nor can it accommodate inductance models, which are required for long low-loss lines. In these cases higher order moments must be used to create an accurate delay approximation.

This chapter begins with an attempt at extending Elmore's distribution function interpretation by fitting the circuit response moments to specific probability density functions (p.d.f.). This includes fitting to double exponential functions which we show in later chapters as corresponding to a two-pole approximation in circuit theory. The chapter concludes with metrics for approximating crosstalk and for controlling the damping, delay, and attenuation in RLC trees.

3.1 Gamma Distribution Model

Elmore's original delay approximation is based on the analogy between non-negative impulse responses and probability density functions (p.d.f.) as described in Chapter 2. In theory, Elmore's distribution interpretation can be extended beyond simply estimating the median by the mean if higher order moments can be used to characterize a

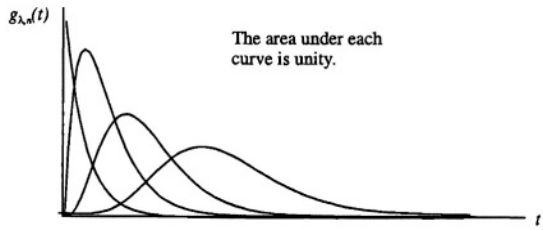


FIGURE 3.1 Examples of several members from the gamma distribution family. Each family member corresponds to specific values of the distribution parameters λ and n.

representative distribution function. Once characterized, the delay can be approximated via table-lookup of the median value for the representative distribution family.

One proposal was to use a gamma distribution function [3.1]. The gamma distribution [3.2], depicted in Fig. 3.1, is a reasonably good representation of RC tree impulse responses since it provides good "coverage" of bell shaped curves which are bounded on the left and exponentially decaying to the right.

The probability density function of gamma distribution, $g_{\lambda,n}(t)$, is a function of one variable t and two parameters λ and n (positive real numbers)

$$g_{\lambda,n}(t) = \frac{\lambda^n t^{n-1} e^{-\lambda t}}{\Gamma(n)}, \qquad t > 0 \tag{3.1}$$

where $\Gamma(x)$ is the gamma function

$$\Gamma(x) = \int_0^\infty y^{x-1} e^{-y} dy, \qquad x > 0 \tag{3.2}$$

Now consider an impulse response, $h(t)$, and assume that it is approximated with a gamma probability density function

$$h(t) = g_{\lambda,n}(t) \tag{3.3}$$

Then the transfer function is given by

$$H(s) = \left(\frac{\lambda}{\lambda + s}\right)^n \tag{3.4}$$

The *s*-domain model denoted by (3.4) may be interpreted as a unique pole with a real number order. Notice that when $n = 1$, the gamma distribution model naturally degrades to the dominant pole model. However, the existence of the parameter n increases the degree of freedom of the model. The first few moments of the transfer function can be expressed easily in terms of the parameters n and λ:

$$\begin{aligned} m_0 &= 1 \qquad m_1 = -\frac{n}{\lambda} \\ m_2 &= \frac{n(n+1)}{2\lambda^2} \qquad m_3 = -\frac{n(n+1)(n+2)}{6\lambda^3} \end{aligned} \tag{3.5}$$

Since the gamma function has only two variables, it can be uniquely characterized by fitting it with two moments. For example, using the second and third equations in (3.5), the parameters λ and n can be obtained from the first and second order circuit moments as

$$\lambda = \frac{m_1}{m_1^2 - 2m_2} \qquad n = \frac{-m_1^2}{m_1^2 - 2m_2} \tag{3.6}$$

In terms of the second central moment μ_2, (3.6) becomes

$$\lambda = \frac{-m_1}{\mu_2} \qquad n = \frac{m_1^2}{\mu_2} \tag{3.7}$$

Recall from the previous chapter that both $-m_1$ (the Elmore delay) and μ_2 are positive numbers for RC trees. Therefore the impulse response approximation in (3.3) with the parameters in (3.7) is always stable.

In the approximation above we have used the first two moments. However, at least three moments are generally required to capture essential waveform response characteristics. Therefore, to match the third moment and capture the skewness of the distribution, we add a third variable, Δ, to include one more degree of freedom. We shift the gamma function by Δ to approximate the impulse response

$$h(t) = g_{\lambda, n}(t - \Delta) \tag{3.8}$$

Thus the transfer function becomes

$$H(s) = e^{-s\Delta}\left(\frac{\lambda}{\lambda + s}\right)^n \tag{3.9}$$

and its moments are given by

$$\begin{aligned} m_1 &= -\frac{n}{\lambda} - \Delta \\ m_2 &= \frac{n(n+1)}{2\lambda^2} + \frac{n}{\lambda}\Delta + \frac{\Delta^2}{2} \\ m_3 &= -\frac{n(n+1)(n+2)}{6\lambda^3} - \frac{n(n+1)}{2\lambda^2}\Delta - \frac{n}{\lambda}\frac{\Delta^2}{2} - \frac{\Delta^3}{6} \end{aligned} \tag{3.10}$$

with $m_0 = 1$. We now have three unknowns, λ, n, and Δ, in three equations. It can be shown that this equation system can also be expressed in terms of m_1, and second and third central moments, μ_2 and μ_3:

$$\begin{aligned} m_1 &= -\frac{n}{\lambda} - \Delta \\ \mu_2 &= \frac{n(n+1)}{\lambda^2} - \left(\frac{n}{\lambda}\right)^2 = \frac{n}{\lambda^2} \\ \mu_3 &= \frac{n(n+1)(n+2)}{\lambda^3} - 3\left(\frac{n}{\lambda}\right)\frac{n(n+1)}{\lambda^2} + 2\left(\frac{n}{\lambda}\right)^3 = \frac{2n}{\lambda^3} \end{aligned} \tag{3.11}$$

Note that the second and third order central moments are independent of the shift, Δ, which is not surprising because μ_2 and μ_3 are the measures of variance and skewness of the waveform, respectively, and they are not affected by the time shift.

Rearranging (3.11) results in

$$\begin{aligned} \lambda &= \frac{2\mu_2}{\mu_3} \\ n &= \frac{4(\mu_2)^3}{(\mu_3)^2} \\ \Delta &= -m_1 - \frac{n}{\lambda} \end{aligned} \tag{3.12}$$

We know from the last chapter that both μ_2 and μ_3 are positive for RC trees. For a causal approximation we also need Δ to be positive. Unfortunately, Δ is not always positive. Therefore, if we get a positive Δ from (3.12), we use (3.8) with (3.12), otherwise we use the simpler approximation in (3.3) with (3.7), which we know is always stable for RC trees.

We now find an expression for the step response, $y(t)$, which is the integral of the impulse response

$$y(t) = \int_0^t h(\tau)d\tau \tag{3.13}$$

Using (3.8) and after some algebraic manipulation we obtain

$$\begin{aligned} y(t) &= \int_0^{\lambda(t-\Delta)} g_{1,n}(\tau)d\tau \\ &= P(n, \lambda(t-\Delta)) \end{aligned} \tag{3.14}$$

where $P(n, x)$ is the incomplete gamma function [3.2]

$$P(n,t) = \frac{1}{\Gamma(n)}\int_0^t x^{n-1}e^{-x}dx \tag{3.15}$$

and $\Gamma(n)$ is the gamma function defined in (3.2).

Note that $P(n, t)$ is zero at $t = 0$ and monotonically increases to one. Hence, to calculate the delay at a particular percentage point α, we only need to find the value of x such that

$$P(n,x) = \alpha \tag{3.16}$$

where $x = \lambda(t-\Delta)$. Then with a simple scaling and shifting, the delay t_α is obtained:

$$t_\alpha = \frac{x}{\lambda} + \Delta \tag{3.17}$$

Therefore all we need is a one dimensional lookup table for the incomplete gamma function percentiles for each predetermined percentage point α.

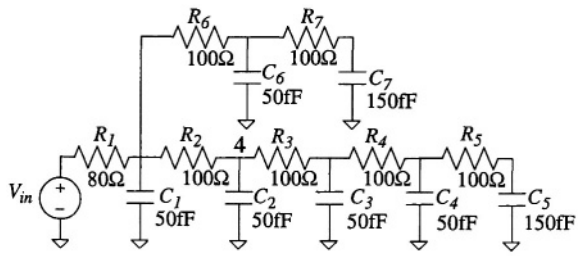


FIGURE 3.2 A simple RC tree circuit.

For example, consider the simple RC tree shown in Figure 3.2. For the voltage response at capacitor C_5, the gamma approximation parameters are found as $n = 1.1595$, $\lambda = 0.0095$ 1/ps, and $\Delta = 12.524$ ps. The corresponding impulse and step response approximations are compared with the actual ones in Figure 3.3. For this voltage response, the 50% gamma approximation delay is found as 101.5 ps, which is a very good approximation of the actual delay, 102.5 ps. The Elmore delay is 134 ps. In this case, since we were able to use the first three moments, we captured the waveshape of the impulse response very accurately.

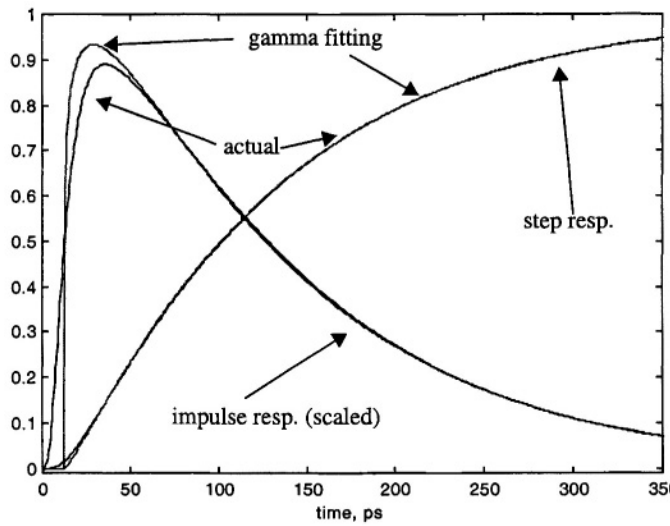


FIGURE 3.3 Approximate and actual voltage impulse and step responses at C_5 for the tree in Figure 3.2. The impulse responses are scaled by 1/7e-3.

But this is not always the case. For example, consider a response at a node closer to the source. At capacitor C_1, $\Delta = -4.78$ ps is negative. From (3.7), we obtain $n = 0.3036$ and $\lambda = 0.0069$ 1/ps by using the first two moments. The approximate and actual impulse responses are shown in Figure 3.4. Clearly, a two-moment approximation is unable to capture the highly unsymmetrical waveshape. In the same figure, the corresponding step responses are also shown. For this node, the gamma, actual, and the Elmore delays are 6.5, 8.0, and 44 picoseconds, respectively.

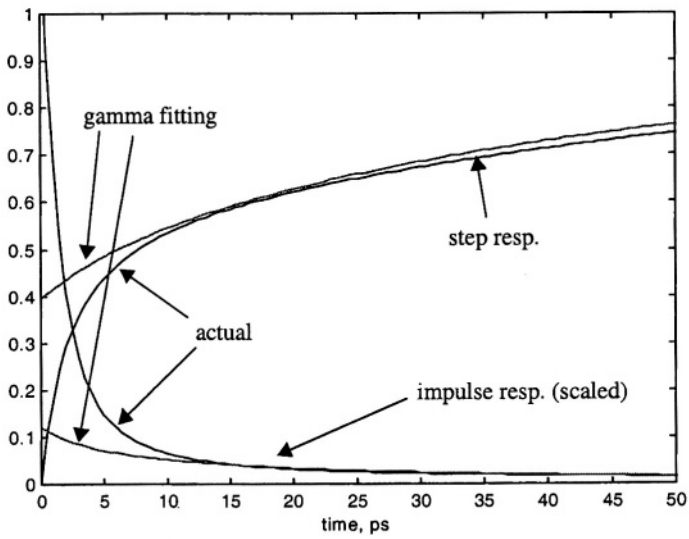


FIGURE 3.4 Approximate and actual impulse and step responses at C_1 for the tree in Figure 3.2. The impulse responses are scaled by 5.

3.2 h-gamma: Gamma Fitting of Homogeneous Response

Instead of approximating the impulse response, we can also fit the gamma distribution to the normalized homogeneous response. As we show next, this approach, which was first proposed in [3.3], gives better results.

Consider a step response, $Y(s)$, and its expression in terms of impulse response, $H(s)$, and impulse response moments:

$$Y(s) = \frac{H(s)}{s} = \frac{m_0}{s} + m_1 + m_2 s + m_3 s^2 + m_4 s^3 + \ldots \tag{3.18}$$

Referring to (3.18), we can decompose the step response into the forced part, m_0/s, and the homogeneous part $m_1 Y_h(s) = m_1 + m_2 s + m_3 s^2 + m_4 s^3 + \ldots$. Therefore, we can write the step response as

$$y(t) = m_0 u(t) + m_1 y_h(t) \tag{3.19}$$

where $y_h(t)$, the inverse Laplace transform of $Y_h(s)$, is the normalized homogeneous response. The components of $y(t)$ are displayed in Figure 3.5.

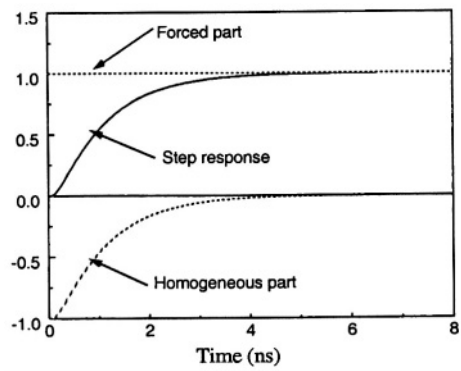


FIGURE 3.5 Decomposition of a unit step response as $u(t)$ and $m_1 y_h(t)$.

In RC trees and meshes, the unit step response, $y(t)$, monotonically increases from zero to one. Furthermore, we know that $m_0 = 1$ and $m_1 < 0$. Thus, the homogeneous response $y_h(t)$ satisfies the conditions of a probability density function:

$$\begin{aligned} y_h(t) &= \frac{1 - y(t)}{-m_1} \geq 0 \\ \int_0^\infty y_h(t)dt &= \frac{1}{-m_1}\int_0^\infty (1 - y(t))dt = 1 \end{aligned} \tag{3.20}$$

where we have used the fact that $\int_0^\infty (1 - y(t))dt = T_D = -m_1$ [3.4].

Therefore, in a similar manner to the gamma distribution modeling of the impulse response, we can treat $y_h(t)$ as a probability density function and approximate it with a gamma distribution:

$$y_h(t) = g_{\lambda, n}(t) \tag{3.21}$$

Now consider the moment expansion for $Y_h(s)$ and $G_{\lambda, n}(s)$:

$$Y_h(s) = 1 + \frac{m_2}{m_1}s + \frac{m_3}{m_1}s^2 + \frac{m_4}{m_1}s^3 + \dots \tag{3.22}$$

$$G_{\lambda, n}(s) = 1 - \frac{n}{\lambda}s + \frac{n(n+1)}{2\lambda^2}s^2 + \dots \tag{3.23}$$

Matching the terms with s and s^2 yields the parameters n and λ in terms of impulse response moments:

$$\begin{aligned} n &= \frac{-m_2^2}{m_2^2 - 2m_1m_3} \\ \lambda &= \frac{m_1m_2}{m_2^2 - 2m_1m_3} \end{aligned} \tag{3.24}$$

We can also express n and λ in terms of the mean, $\mu(y_h)$, and the variance, $\mu_2(y_h)$, of $y_h(t)$:

$$\begin{aligned} n &= \frac{\mu^2}{\mu_2} \\ \lambda &= \frac{\mu}{\mu_2} \end{aligned} \tag{3.25}$$

Equation (3.25) can easily be verified by considering the relations

$$\begin{aligned} \mu &= \frac{-m_2}{m_1} \\ \mu_2 &= 2\left(\frac{m_3}{m_1}\right) - \left(\frac{m_2}{m_1}\right)^2 \end{aligned} \tag{3.26}$$

which simply follow from the definitions in Subsection 2.5.1.

Once the parameters n and λ are obtained from the circuit moments, the approximate step response is

$$y(t) = m_0 u(t) + m_1 g_{\lambda,n}(t) \tag{3.27}$$

It is evident from (3.25) and (3.26) that we are using the first three moments of the impulse response. Moreover, by fitting the complete response with the summation of the forced response and the gamma distribution approximation, it is apparent from (3.18) that the first three moments of the impulse response are implicitly matched.

Consider the RC tree circuit shown in Figure 3.2 again. The step response approximation given in (3.27) and the actual response at C_1 are plotted in Figure 3.6. A comparison of the step response approximations in Figure 3.4 and Figure 3.6 clearly shows the advantage of h-gamma fitting. The h-gamma method, by fitting the first three moments, easily captures even the "difficult" responses such as the one at C_1.

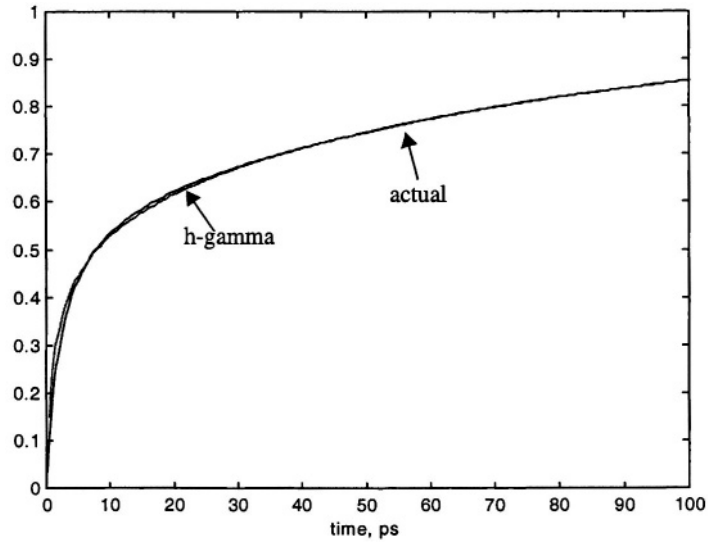


FIGURE 3.6 h-gamma approximated and actual step responses at C_1 for the tree in Figure 3.2.

A. Delay Calculation

To find the step response delay for any percentage point α, we have to find the value of t in (3.27) which satisfies

$$\alpha = 1 + \frac{m_1\lambda(\lambda t)^{n-1}e^{-\lambda t}}{\Gamma(n)} \tag{3.28}$$

where we have substituted (3.1) for $g_{\lambda,n}(t)$ and assumed $m_0 = 1$. By defining the following two parameters

$$k = -m_1\lambda\,,\; x = \lambda t \tag{3.29}$$

and using the expression

$$y_{k,n}(x) = 1 - \frac{kx^{n-1}e^{-x}}{\Gamma(n)} \tag{3.30}$$

we can rewrite (3.28) as

$$\alpha = y_{k,n}(x) \tag{3.31}$$

Instead of solving for t using the nonlinear expression in (3.28), we use a table lookup approach to evaluate (3.31). For each predetermined percentage point α, we precompile a 2-dimensional table with k and n as its entries, and x as its output. Thus when n and λ are obtained by matching the first three moments, we compute k in (3.29), and use k and n to get x via table lookup. The delay value, t, is then obtained by scaling x back by λ, i.e., $t = x/\lambda$.

The response of a saturated ramp with a rise time t_r can be expressed in terms of integration of the step response,

$$r(t) = \begin{cases} \dfrac{1}{t_r}\displaystyle\int_0^t y(\tau)d\tau & t \le t_r \\[2ex] \dfrac{1}{t_r}\left(\displaystyle\int_0^t y(\tau)d\tau - \int_0^{t-t_r} y(\tau)d\tau\right) & t > t_r \end{cases} \tag{3.32}$$

Defining

$$l = \frac{t_r}{-m_1} \tag{3.33}$$

we can express the ramp response as

$$r(t) = \begin{cases} \dfrac{1}{kl}\displaystyle\int_0^x y_{k,n}(\tau)\,d\tau & x \le kl \\ \dfrac{1}{kl}\left(\displaystyle\int_0^x y_{k,n}(\tau)\,d\tau - \int_0^{x-kl} y_{k,n}(\tau)\,d\tau\right) & x > kl \end{cases} \tag{3.34}$$

Following the approach outlined for the step delay evaluation, we can write $r(t)$ in (3.34) as $r_{k,l,n}(x)$. Hence, for a given percentage point α, the delay t can be obtained by finding the value of x which satisfies

$$\alpha = r_{k,l,n}(x) \tag{3.35}$$

A 3-dimensional table with entries k, l, and n, and with output x, can be precompiled for the solution of (3.35). As a summary, the saturated ramp delay computation steps include the calculation of k, l, and n, then using the 3-dimensional table to get x, and then scaling it by λ to get t. The delay relative to the input ramp is then calculated as $t - \alpha T_r$.

B. Stability Issues

To have a valid (stable) gamma distribution delay model, the parameters n and λ should be positive. We now show that the gamma distribution model for the homogeneous portion of a step response is stable for any RC mesh/tree.

Any step response in an RC mesh or tree monotonically increases from zero to one, i.e., $y(0) = 0, y(\infty) = 1$. Consequently, the normalized homogeneous response, $y_h(t)$ monotonically decreases from $1/(-m_1)$ to zero. Therefore it is bounded and always positive. Since its mean μ and variance μ_2 are defined as

$$\mu = \int_0^\infty t y_h(t)dt$$
$$\mu_2 = \int_0^\infty (t-\mu)^2 y_h(t)dt \tag{3.36}$$

it is obvious that μ and μ_2 are always positive for the homogeneous response $y_h(t)$. From (3.25), it follows that the h-gamma parameters always satisfy $n > 0$, $\lambda > 0$ for RC meshes and trees.

C. Properties of the Tables

One important practical issue with the h-gamma method is the runtime and storage efficiency properties of the table model. Note that of the three parameters, k and n are not associated with the input ramp rise time, and can be expressed in terms of the circuit-response moments. From (3.25) and (3.29), we have

$$k = \frac{m_1^2 m_2}{2m_3 m_1 - m_2^2}$$
$$n = \frac{m_2^2}{2m_3 m_1 - m_2^2} \tag{3.37}$$

To determine the expected ranges for these values, first consider the case when the response is dominated by a single pole p, such that $m_1 \approx m_2/m_1 \approx m_3/m_2 \approx 1/p$. Using this relation it is apparent from (3.37) that k and n should be close to 1. Empirical results validate this, where it is observed that k is between 0.3~1.8, and n is between 0.5~1.5 [3.3].

To determine the range of l it should be noted that when $l > 10$, i.e, $t_r > 10(-m_1)$, the response will be a ramp follower and the 50% percentage delay point will approach the Elmore delay upper-bound. When $l < 0.1$, i.e., $t_r < 0.1(-m_1)$, the ramp is steep enough to be considered as a step input, which means the delay can be approximated from the lookup table for step response delay with entries k and n. Thus we only need to make the table with the value of l in the range of 0.1~10.

D. Example

This example is taken from [3.3]. It considers a circuit from a 0.25 micron technology commercial microprocessor. The RC tree has 50 fanouts, including nodes close to the driver as well as nodes far from the driving point. The 50% delay numbers are compared for three different methods: (a) the Elmore delay; (b) the time-shifted gamma method, presented in Section 3.1, which fits the gamma function to the impulse response; (c) and the h-gamma method. The input rise time is 0.2 times that of the Elmore delay.

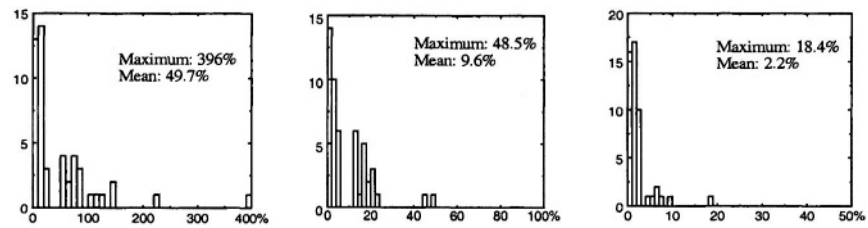


FIGURE 3.7 The 50% delay errors in (a) the Elmore metric; (b) the time-shifted gamma method; (c) and the h-gamma method, for an RC tree with 50 fanouts.

3.3 Double Exponential Impulse Response Distribution

The dominant pole model that was presented in the previous chapter, is based on the approximation of the RC tree step response in terms of a single time-constant model,

$$v(t) = (1 - e^{p_D t})u(t) \tag{3.38}$$

where the pole p_D is the inverse of the Elmore delay, $p_D = -1/T_D$.

The corresponding impulse response is given by

$$h(t) = -p_D e^{p_D t} u(t) \tag{3.39}$$

with the *s*-domain representation of

$$H(s) = \frac{-p_D}{s - p_D} \tag{3.40}$$

Importantly, this single pole model assumes that there are no low-frequency zeros and that there exists a single pole which dominates the low-frequency behavior of the circuit. However, with the increasing effect of the wiring resistance, these assumptions can break down such that more than one pole is required to accurately capture the delay. This is illustrated in Figure 3.8, in which we compare the single-pole step response approximation with the actual one at C_1 for the RC tree shown in Figure 3.2.

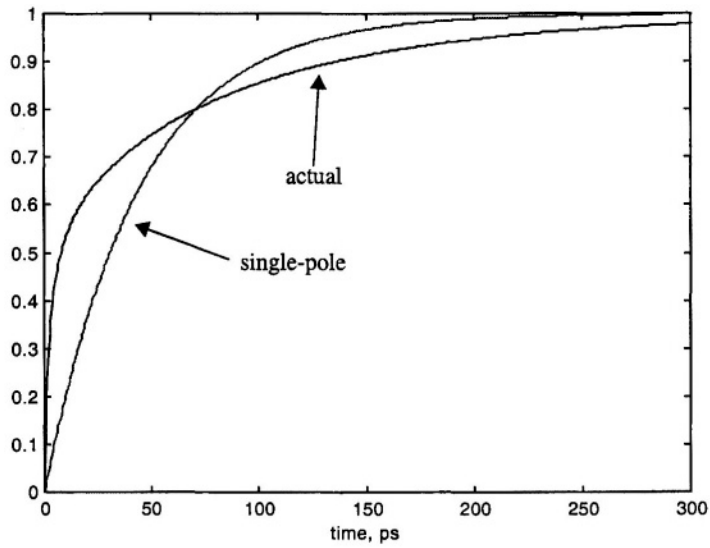


FIGURE 3.8 Single-pole model and actual step responses at C_1 for the tree in Figure 3.2.

The most obvious next step would be a model with two dominant poles. In terms of our p.d.f. interpretation of impulse response delay modeling this would correspond to a double exponential distribution function:

$$h(t) = k_1 e^{p_1 t} + k_2 e^{p_2 t} \tag{3.41}$$

where $p_1 \neq p_2$. In the Laplace domain, it is represented by a two-pole one-zero transfer function

$$H(s) = \frac{k_1}{s-p_1} + \frac{k_2}{s-p_2} = \frac{K(s-z)}{(s-p_1)(s-p_2)} \tag{3.42}$$

The delay can then be obtained explicitly from the corresponding step or saturated ramp response using a table-lookup method as we describe later in this section.

One of the first approaches to characterize transfer functions using two poles was proposed in [3.5]. It approximates the original transfer function of an RC tree with a two-pole one-zero transfer function. Rewrite (3.42) in the form

$$H(s) = \frac{1+s\tau_z}{(1+s\tau_1)(1+s\tau_2)} \tag{3.43}$$

and consider its Taylor series expansion about $s = 0$

$$H(s) = 1 + (\tau_z - \tau_p)s + \tau_p(\tau_p - \tau_z - \tau_1\tau_2/\tau_p)s^2 + \dots \tag{3.44}$$

where $\tau_p = \tau_1 + \tau_2$. The three parameters in (3.43) are determined by matching the first two moments of the exact impulse response (at node *e*),

$$H(s) = 1 - sT_{De} + s^2\sum_k R_{ke}C_kT_{Dk} + \dots \tag{3.45}$$

where T_{Di} is the Elmore delay at node *i*, R_{ki} is the resistance of the portion of the (unique) path between the input and node *i*, that is common with the (unique) path between the input and node *k*, and C_k is the capacitance at node *k*.

Instead of using higher order moments, the model approximates τ_P by the sum of all open-circuit time constants in the original circuit

$$\tau_p = \sum_k R_{kk}C_k \tag{3.46}$$

Next, a two-pole approximation is described which is obtained by matching the first three moments.

3.3.1 Two-Pole Approximation With Explicit Moment Matching

Consider a transfer function $H(s)$ in an RC tree circuit, and assume that a sufficient number of its moments are calculated from the circuit:

$$H(s) = 1 + m_1 s + m_2 s^2 + \ldots \tag{3.47}$$

With a two-pole one-zero approximation as given in (3.42), the transfer function can be expressed as a ratio of two polynomials

$$H(s) = \frac{1 + b_1 s}{1 + a_1 s + a_2 s^2} \tag{3.48}$$

where we have used the fact that $H(0) = 1$ for RC trees. The Taylor series expansion of the function $H(s)$ about $s = 0$ can be expressed in terms of its coefficients as

$$\begin{aligned} H(s) = {} & 1 + (b_1 - a_1)s + [-a_2 - a_1(b_1 - a_1)]s^2 + \\ & [a_1(a_2 + a_1(b_1 - a_1)) - a_2(b_1 - a_1)]\, s^3 + \ldots \end{aligned} \tag{3.49}$$

By matching the coefficients of s through s^3 in (3.47) and (3.49) and after some algebraic manipulation, we obtain the following linear equation system for the coefficients in (3.48):

$$\begin{aligned} b_1 &= a_1 + m_1 \\ 0 &= a_2 + a_1 m_1 + m_2 \\ 0 &= a_2 m_1 + a_1 m_2 + m_3 \end{aligned} \tag{3.50}$$

From the last two equations in (3.50), the coefficients of the denominator polynomial are given by

$$\begin{aligned} a_1 &= \frac{m_1 m_2 - m_3}{m_2 - m_1^2} \\ a_2 &= \frac{m_1 m_3 - m_2^2}{m_2 - m_1^2} \end{aligned} \tag{3.51}$$

Once a_1 and a_2 are known, the poles, p_1 and p_2, which are the roots of the denominator polynomial in (3.48), are found easily. To find the residues of the poles, we match the two-pole model to the first two moments of the actual transfer function. Consider the moment expansion of the two-pole transfer function,

$$H(s) = \frac{k_1}{s-p_1} + \frac{k_2}{s-p_2} = -\frac{k_1}{p_1} - \frac{k_2}{p_2} + s\left(-\frac{k_1}{p_1^2} - \frac{k_2}{p_2^2}\right) + \cdots \tag{3.52}$$

The moment matching equations are

$$\begin{aligned} -\frac{k_1}{p_1} - \frac{k_2}{p_2} &= m_0 = 1 \\ -\frac{k_1}{p_1^2} - \frac{k_2}{p_2^2} &= m_1 \end{aligned} \tag{3.53}$$

Thus

$$k_1 = \frac{p_1^2(1-m_1p_2)}{p_2-p_1} \qquad k_2 = \frac{p_2^2(-1+m_1p_1)}{p_2-p_1} \tag{3.54}$$

Now consider again the RC tree example shown in Figure 3.2. Figure 3.9 compares the two-pole step response approximations at C_1 and C_5 with the actual ones. As seen from the figure, the two-pole model is a very good approximation for the "difficult" response at C_1. And as we move away from the source, the two-pole model becomes a perfect approximation.

This two-pole approximation using moment matching can be extended beyond the fitting of p.d.f.'s, and can be applied for any number of poles using asymptotic waveform evaluation (AWE), which is described in detail in Chapter 4. One difficulty that the aforementioned double exponential model and AWE share is the realizability of the moment fitting. For example, moments from a higher order system are not guaranteed to produce a stable 2-pole model via the fitting process described above. Chapter 4 describes ways of overcoming this limitation for AWE, but here we describe two provably stable double exponential models that can provide an explicit delay metric. But first we address the stability problem in 2-pole models.

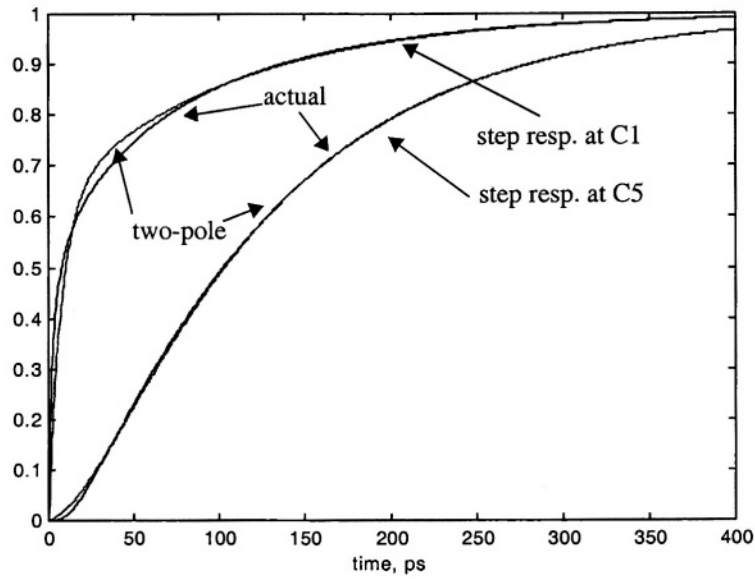


FIGURE 3.9 Two-pole model and actual step responses at C_1 and C_5 for the tree in Figure 3.2.

3.3.2 Stability of Two-Pole Approximation

For a stable two-pole model, the roots of the denominator polynomial

$$Q(s) = 1 + a_1 s + a_2 s^2 \tag{3.55}$$

in (3.48) must have negative real parts.

To investigate the stability properties of two-pole models we will use the Routh stability criteria: *The second order polynomial Q(s) given in (3.55) is stable if and only if the coefficients* a_1 *and* a_2 *are positive.*

Now let us rewrite the coefficients in (3.51) in the form

$$a_1 = \frac{-m_1(-m_2 + m_3/m_1)}{m_2 - m_1^2} \qquad a_2 = \frac{m_1^2(m_3/m_1 - m_2^2/m_1^2)}{m_2 - m_1^2} \tag{3.56}$$

It can be shown that if the conditions

$$m_2 > m_1^2 \quad \text{and} \quad m_3 < \frac{m_2^2}{m_1}$$
$$\text{or} \tag{3.57}$$
$$m_2 < m_1^2 \quad \text{and} \quad m_3 > \frac{m_2^2}{m_1}$$

are satisfied, then the coefficients a_1 and a_2 in (3.56) are positive. A little later we show that this is always the case for admittance and impedance functions in RC circuits. But RC transfer function moments do not satisfy (3.57) in general. They have very similar bounds though, as we will see next.

In the previous chapter we showed that the second and third central moments of RC tree responses are always positive, that is,

$$\begin{aligned} \mu_2 &= 2m_2 - m_1^2 > 0 \\ \mu_3 &= -6m_3 + 6m_1 m_2 - 2m_1^3 > 0 \end{aligned} \tag{3.58}$$

Thus, RC transfer function moments satisfy

$$\begin{aligned} m_2 &> \frac{m_1^2}{2} \\ m_3 &< m_1 m_2 - \frac{m_1^3}{3} \end{aligned} \tag{3.59}$$

In order to compare the stability requirements in (3.57) with the actual bounds in (3.59), we normalize the moments as

$$\bar{m}_i = m_i / (-m_1)^i . \tag{3.60}$$

This is equivalent to scaling the time by $-m_1$, the Elmore delay. In terms of normalized moments, the stability requirements in (3.57) become

$$\bar{m}_2 > 1 \text{ and } -\bar{m}_3 > \bar{m}_2^2$$
$$\text{or} \tag{3.61}$$
$$\bar{m}_2 < 1 \text{ and } -\bar{m}_3 < \bar{m}_2^2$$

Similarly, the RC moment bounds can be simplified as

$$\frac{1}{2} < \bar{m}_2 < -\bar{m}_3 + \frac{1}{3} \tag{3.62}$$

The regions defined by (3.61) and (3.62) are depicted in Figure 3.10. The shaded area is the stable region. The triangular region described by (3.62) is shown by dashed lines. Thus a moment pair that is outside of the shaded area gives an unstable 2-pole model. But, empirical results have shown that RC moments usually, but not always, satisfy the stability conditions given in (3.61) [3.6].

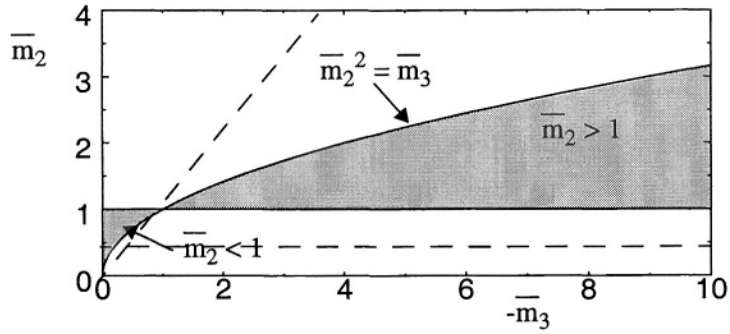


FIGURE 3.10 Stability requirements and actual RC moment bounds.

3.3.3 A Stable 2-pole Model Based on First Three Moments

We now present a method which uses the first three moments and provides a provably stable 2-pole model [3.7]. We first explain how we find the dominant pole. Then we describe how the second pole can be approximated using the first pole and the first three moments.

Consider an RC tree circuit with n distinct poles and let $H(s)$ be a response in the circuit

$$H(s) = \sum_{i=1}^{n} \frac{k_i}{s - p_i} = 1 + m_1 s + m_2 s^2 + \ldots \tag{3.63}$$

The expression of the *j*th moment in terms of poles and residues is

$$m_j = -\sum_{i=1}^{n} \frac{k_i}{p_i^{j+1}} \tag{3.64}$$

Assume that the poles are ordered such that $-p_1 < -p_2 < \ldots < -p_n$ and consider the ratio of two successive moments

$$\frac{m_j}{m_{j+1}} = \frac{p_1\left(k_1 + k_2\left(\frac{p_1}{p_2}\right)^{j+1} + \ldots + k_n\left(\frac{p_1}{p_n}\right)^{j+1}\right)}{k_1 + k_2\left(\frac{p_1}{p_2}\right)^{j+2} + \ldots + k_n\left(\frac{p_1}{p_n}\right)^{j+2}} \tag{3.65}$$

It follows that

$$\lim_{j \to \infty} \frac{m_j}{m_{j+1}} = p_1 \tag{3.66}$$

Therefore, using only the information contained in the first three impulse response moments, the most accurate approximation of the first pole is

$$p_1 = \frac{m_2}{m_3} \tag{3.67}$$

From (3.59) it is easy to show that for RC tree responses $m_2 > 0$ and $m_3 < 0$. Thus p_1 is always negative.

To explain how we arrive at a guaranteed stable approximation for the second pole, we begin with the ratio of two successive moments in (3.65). Using polynomial division, we can rewrite (3.65) as

$$\frac{m_j}{m_{j+1}} = p_1\left[1 + \frac{k_2}{k_1}\left(\frac{p_1}{p_2}\right)^{j+1}\left(1 - \frac{p_1}{p_2}\right) + \frac{k_3}{k_1}\left(\frac{p_1}{p_3}\right)^{j+1}\left(1 - \frac{p_1}{p_3}\right) + \ldots\right] \tag{3.68}$$

Next we consider the following ratio of terms:

$$\frac{\left(\frac{m_j}{m_{j+1}} - \frac{m_{j+1}}{m_{j+2}}\right)}{\left(\frac{m_{j+1}}{m_{j+2}} - \frac{m_{j+2}}{m_{j+3}}\right)} \tag{3.69}$$

Using (3.68), the ratio in (3.69) becomes

$$\frac{\left(\frac{m_j}{m_{j+1}} - \frac{m_{j+1}}{m_{j+2}}\right)}{\left(\frac{m_{j+1}}{m_{j+2}} - \frac{m_{j+2}}{m_{j+3}}\right)} = \frac{\frac{k_2}{k_1}\left(\frac{p_1}{p_2}\right)^{j+1}\left(1 - \frac{p_1}{p_2}\right)^2 + \frac{k_3}{k_1}\left(\frac{p_1}{p_3}\right)^{j+1}\left(1 - \frac{p_1}{p_3}\right)^2 + \ldots}{\frac{k_2}{k_1}\left(\frac{p_1}{p_2}\right)^{j+2}\left(1 - \frac{p_1}{p_2}\right)^2 + \frac{k_3}{k_1}\left(\frac{p_1}{p_3}\right)^{j+2}\left(1 - \frac{p_1}{p_3}\right)^2 + \ldots} \tag{3.70}$$

Note that the right hand side of (3.70) approaches p_2/p_1 as j increases, that is,

$$\lim_{j \to \infty} \frac{\left(\frac{m_j}{m_{j+1}} - \frac{m_{j+1}}{m_{j+2}}\right)}{\left(\frac{m_{j+1}}{m_{j+2}} - \frac{m_{j+2}}{m_{j+3}}\right)} = \frac{p_2}{p_1} \tag{3.71}$$

However, due to the uncertainty of the monotonicity of the moment ratios, we do not know the sign of (3.69), but we can still consider convergence to the exact magnitude of the ratio of the first two poles by taking the absolute values of both sides in (3.71).

Therefore, based on the first three moments, our most accurate approximation for the second pole is

$$p_2 = p_1\left|\frac{\left(\frac{m_0}{m_1} - \frac{m_1}{m_2}\right)}{\left(\frac{m_1}{m_2} - \frac{m_2}{m_3}\right)}\right| \tag{3.72}$$

With a pair of stable poles that approximates the first two poles of the actual circuit, we can find the residues in

$$H(s) = \frac{k_1}{s-p_1} + \frac{k_2}{s-p_2} \tag{3.73}$$

as explained in Subsection 3.3.1.

3.3.4 Stable 2-Pole Method (S2P)

The *stable two pole* (S2P) method constructs the approximate transfer function given in (3.42) in two stages [3.8]:

1. Find the poles using driving point admittance moments.
2. Obtain the residues at the desired outputs using the poles found in the first stage and the transfer function moments at the output of interest. Note that the approximate driving point poles are common to all outputs. So they are calculated only once for a given circuit.

We now show that the 2-pole approximation always gives stable poles when it is applied to driving point immittance function in RC trees.

First consider a driving point admittance function $Y(s)$ and its moment expansion:

$$Y(s) = y_0 + y_1 s + y_2 s^2 + \ldots \tag{3.74}$$

Now consider a second order approximation for $Y(s)$ in the form

$$Y(s) = k_0 + \frac{k_1}{s-p_1} + \frac{k_2}{s-p_2} \tag{3.75}$$

Note that the approximation in (3.75) is a two-pole two-zero system, and thus it is slightly different than the one given in (3.42). $Y(s)$ can also be expressed as

$$Y(s) = \frac{b_0 + b_1 s + b_2 s^2}{1 + a_1 s + a_2 s^2} \tag{3.76}$$

Expanding $Y(s)$

$$Y(s) = b_0 + (b_1 - b_0 a_1)s + (b_2 - b_0 a_2 - a_1(b_1 - b_0 a_1))s^2 + \ldots \tag{3.77}$$

and by matching the first five terms in (3.74) and (3.77) and after some straightforward manipulations, we end up with a linear equation system for the coefficients:

$$\begin{aligned} b_0 &= y_0 \\ b_1 &= a_1 y_0 + y_1 \\ b_2 &= a_2 y_0 + a_1 y_1 + y_2 \\ 0 &= a_2 y_1 + a_1 y_2 + y_3 \\ 0 &= a_2 y_2 + a_1 y_3 + y_4 \end{aligned} \tag{3.78}$$

To find the poles of $Y(s)$ we only need the coefficients of the denominator polynomial. From the last two equations we obtain

$$a_1 = \frac{y_2 y_3 - y_1 y_4}{y_1 y_3 - y_2^2}, \qquad a_2 = \frac{y_2 y_4 - y_3^2}{y_1 y_3 - y_2^2} \tag{3.79}$$

As mentioned in the previous section, the above procedure is equivalent to finding a second order AWE approximation to the input admittance function. In the next chapter, we show that to find the coefficients of the denominator polynomial, we can use any four successive moments. Sometimes using higher order moments can give better approximations to the original poles. Therefore, for generality we rewrite (3.79) as

$$a_1 = \frac{y_{i+1} y_{i+2} - y_i y_{i+3}}{y_i y_{i+2} - y_{i+1}^2}, \qquad a_2 = \frac{y_{i+1} y_{i+3} - y_{i+2}^2}{y_i y_{i+2} - y_{i+1}^2}, \qquad i > 0 \tag{3.80}$$

Recall from the Hurwitz criteria that the denominator polynomial in (3.76) is stable if and only if the coefficients given in (3.80) are positive. To prove that this is the case for RC tree admittance functions, we will make use of the following two properties.

Property 1: *The moments of* $Y(s)$ *satisfy*

$$\begin{aligned} y_i &< 0 \qquad \text{if } i = 2, 4, \ldots \\ y_i &> 0 \qquad \text{if } i = 1, 3, \ldots \end{aligned} \tag{3.81}$$

The proof is given in Appendix 2.A.

Property 2: *The ratio of successive moments of the driving point admittance function in an RC network is monotonically increasing, i.e.,*

$$\frac{y_{i-1}}{y_i} < \frac{y_i}{y_{i+1}} \qquad i > 1 \tag{3.82}$$

The proof is given in Appendix 3.A.1.

The coefficients in (3.80) can be rewritten as

$$a_1 = -\frac{y_{i+3}}{y_{i+2}} \frac{\left(\frac{y_i}{y_{i+1}} - \frac{y_{i+2}}{y_{i+3}}\right)}{\left(\frac{y_i}{y_{i+1}} - \frac{y_{i+1}}{y_{i+2}}\right)}, \qquad a_2 = \frac{y_{i+3}}{y_{i+1}} \frac{\left(\frac{y_{i+1}}{y_{i+2}} - \frac{y_{i+2}}{y_{i+3}}\right)}{\left(\frac{y_i}{y_{i+1}} - \frac{y_{i+1}}{y_{i+2}}\right)} \tag{3.83}$$

Since $-y_{i+3}/y_{i+2}$ and y_{i+3}/y_{i+1} are always positive, and using the monotonic convergence property of moments one can show that

$$a_1 > 0, \qquad a_2 > 0 \tag{3.84}$$

Therefore, the second order approximation for driving point admittance function of an RC circuit is always stable. Furthermore, in Appendix 3.A.2 we prove that $a_1^2 - 4a_2 > 0$. Thus, the poles are also real numbers.

Once the approximate driving point poles are obtained, a transfer function can be formed for any node in the circuit:

$$H(s) = \frac{k_1}{s - p_1} + \frac{k_2}{s - p_2} \tag{3.85}$$

The residues in (3.85) are calculated by matching the first two moments, m_0 and m_1, of the response at the output node as explained in Subsection 3.3.1:

$$k_1 = \frac{m_1 p_2 - m_0}{p_1 - p_2} p_1^2 \qquad k_2 = \left(-m_0 - \frac{k_1}{p_1}\right) p_2 \tag{3.86}$$

Now consider again our RC tree circuit shown in Figure 3.2. The S2P step responses at C_1 and C_5 together with the actual ones are given in Figure 3.11. As expected, the S2P is slightly less accurate then the regular two-pole model (Figure 3.9). The reason

is S2P uses the driving point moments to calculate the poles, and then the same poles are used for all of the responses. At a specific response we match only the first two moments exactly. In the standard two-pole model, on the other hand, for each response we match the first four moments (including m_0) to find the poles.

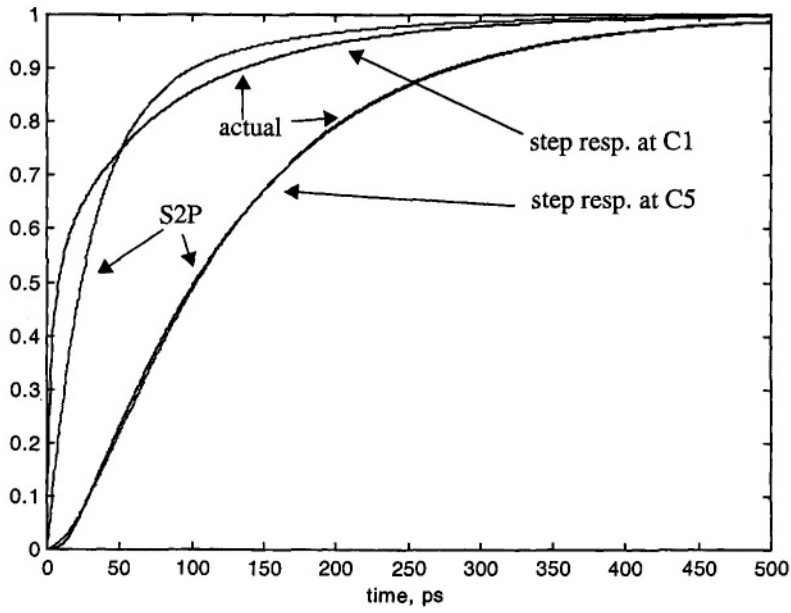


FIGURE 3.11 S2P and actual step responses at C_1 and C_5 for the tree in Figure 3.2.

We conclude this subsection with a brief summary of the S2P method:

1. Compute y_0, y_1, y_2, y_3, y_4 for driving point current.
2. Compute m_0, m_1 at the response nodes.
3. Find two poles at the driving point:

$$p_1 = \frac{-a_1 + \sqrt{a_1^2 - 4a_2}}{2a_2} \qquad p_2 = \frac{-a_1 - \sqrt{a_1^2 - 4a_2}}{2a_2} \tag{3.87}$$

   where the coefficients a_1 and a_2 are obtained using(3.79).
4. For each response node find the residues using (3.86).

3.3.5 Delay Calculation in Two-Pole Models

All the two-pole models mentioned in this section approximate the impulse response in the form

$$h(t) = k_1 e^{p_1 t} + k_2 e^{p_2 t} \tag{3.88}$$

The corresponding step response is given by

$$v_{\text{step}}(t) = \frac{k_1}{p_1}(e^{p_1 t} - 1) + \frac{k_2}{p_2}(e^{p_2 t} - 1) \tag{3.89}$$

The saturated ramp response becomes

$$v(t) = \frac{1}{t_r}\sum_{i=1}^{2} \frac{k_i}{p_i^2}[(e^{p_i t} - 1 - p_i t)u(t) - (e^{p_i (t-t_r)} - 1 - p_i (t-t_r))u(t-t_r)] \tag{3.90}$$

where t_r is the rise time. The delay for an α percentage can be found by solving $v(t) = \alpha$ using Newton-Raphson or regula-falsi iterations. When started with a good initial guess, these methods converge very quickly. We can compute an initial guess using the dominant pole in a one-pole model [3.7].

Alternatively, a table lookup approach can be used to avoid nonlinear iterations. We can further reduce the number of parameters by normalizing the response with respect to one of the parameters. This also yields predictable ranges for the remaining parameters. The parameters for the table would then be t_r/p_1, m_1/p_1 and p_2/p_1, where p_1 is the pole with smaller amplitude. The delay value obtained from the table is then divided by p_1 for back-normalization. Considering the first moment as an input parameter instead of the residues makes it easier to predict the ranges. For example, it is empirically known that after $t_r > 7p_1$, the 50% delay is almost equal to the Elmore delay, $-m_1$, so in the table the range for t_r/p_1 should be between 0 and 7.

3.4 Closed Form RC Delay Metrics

Another alternative for an efficient delay metric is by curve fitting in terms of the first three moments [3.6]. In this approach, a simple closed-form expression is developed by exploiting the properties of RC tree moments:

$$t_d = T_D \frac{\bar{m}_2^2(-\bar{m}_3 b_1 + b_2 - (b_3/\bar{m}_3))}{-\bar{m}_3 a_1 + a_2 - (a_3/\bar{m}_3)} \tag{3.91}$$

where T_D is the Elmore delay and, $\bar{m}_2$ and $\bar{m}_3$ are the normalized second and third moments as defined in (3.60). The universal constants, a_i's and b_i's, are obtained with least squares approximations using stable two-pole approximations.

We now mention a few other closed-form metrics which use the first two moments. The first method [3.9] finds a two-pole model by using the first two moments and adding a time-derivative constraint. The resultant delay expression is

$$t_d = \frac{1}{2}(-m_1 + \sqrt{4m_2 - 3m_1^2}) \ln\left(1 - \frac{m_1}{\sqrt{4m_2 - 3m_1^2}}\right) \tag{3.92}$$

The second method [3.10] finds one pole using two moments and approximates the delay as

$$t_d = \sqrt{2m_2 - m_1^2} \ln(2) \tag{3.93}$$

And finally, the third metric [3.11] is obtained by scaling the dominant time constant delay as

$$t_d = -m_1 \ln(2) \sqrt{\frac{m_1^2}{m_2}} = \frac{m_1^2}{\sqrt{m_2}} \ln(2) \tag{3.94}$$

3.5 Noise Metrics

Delay is not the only quality measure that one is interested in when using interconnect metrics for optimization. Due to dominant coupling capacitance, noise is generally equally important as delay. Moments can be used to estimate crosstalk induced noise in RC interconnect circuits. After addressing noise metrics in this section, we continue with delay and signal integrity issues in RLC interconnect in the following sections.

3.5.1 Crosstalk Noise

The crosstalk noise is caused by the capacitive coupling between a net and a set of aggressor nets. Consider the coupled system shown in Figure 3.12. For simplicity only one aggressor is considered. Since we assume linear(ized) gate models, it is straightforward to extend the results that are presented in this section to the case with multiple aggressors. In Figure 3.12, the aggressor net is excited with a voltage source and the victim net's driver is replaced with a resistor that represents the transistors that are driving the victim line. Let $n(t)$ be the voltage of the output node in the victim net. We present two methods to estimate $N_{\max}$, the maximum value of $n(t)$.

Let $H(s)$ be the transfer function from the input to the output. Assume that it has an exact pole-residue representation in the form

$$H(s) = \sum_{i=1}^{n} \frac{k_i}{s - p_i} \tag{3.95}$$

Consider the moments of the transfer function,

$$H(s) = m_0 + m_1 s + m_2 s^2 + \ldots \tag{3.96}$$

The first two moments can be expressed in terms of poles and residues as

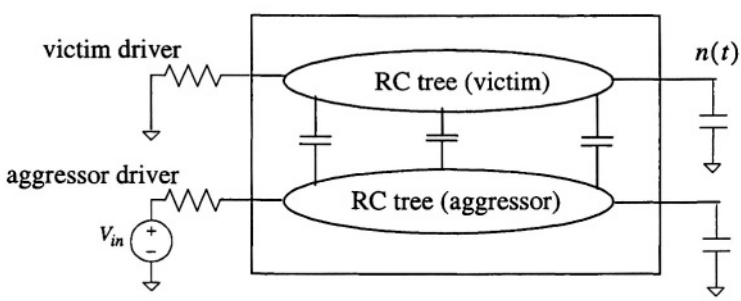


FIGURE 3.12 A circuit containing two capacitively coupled nets.

$$m_0 = -\sum_{i=1}^{n} \frac{k_i}{p_i} \qquad m_1 = -\sum_{i=1}^{n} \frac{k_i}{p_i^2} \tag{3.97}$$

Since there is no dc path between the two nets, we have

$$H(0) = 0 \tag{3.98}$$

which also implies $m_0 = 0$.

For an input waveform $v_{in}(t)$ ($V_{in}(s)$, in the *s*-domain), the noise waveform is given by

$$n(t) = L^{-1}\{V_{in}(s)H(s)\} \tag{3.99}$$

where L^{-1} indicates the inverse Laplace transformation.

3.5.2 First Moment Noise Metric

Now assume that the input is a ramp waveform with a slope of $1/t_r$, i.e., $v_{in}(t) = t/t_r$. Then using (3.95) and (3.99), the noise waveform is given by

$$n(t) = \frac{1}{t_r}\left(\sum_{i}^{n} \frac{k_i}{p_i^2} e^{p_i t} - \sum_{i}^{n} \frac{k_i}{p_i^2} - \sum_{i}^{n} \frac{k_i}{p_i} t\right) u(t) \tag{3.100}$$

Inserting (3.97) into (3.100) and using $m_0 = 0$, we obtain

$$n(t) = \frac{1}{t_r}\left(m_1 + \sum_{i}^{n} \frac{k_i}{p_i^2} e^{p_i t}\right) u(t) \tag{3.101}$$

Since the poles are negative in RC circuit responses, $n(t)$ is bounded, and it can be easily shown that

$$n(0) = 0 \qquad \lim_{t \to \infty} n(t) = \frac{m_1}{t_r} \tag{3.102}$$

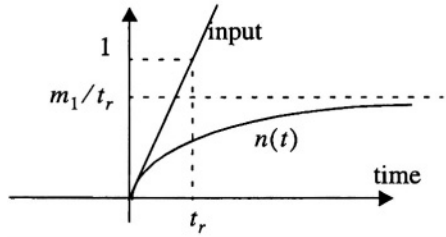


FIGURE 3.13 Noise waveform in case of a ramp input

Now assume that $n(t)$ is monotonic which suggests a noise waveform as shown in Figure 3.13. In this case, it is apparent that the noise is bounded by the steady-state value:

$$N_{\max} = \frac{m_1}{t_r} \tag{3.103}$$

The first moment metric, expressed above, is very simple, however, there are two issues associated with it. First, although most noise responses exhibit a waveform which is similar to the one drawn in Figure 3.13, monotonicity is not guaranteed for general RC circuits. Therefore m_1/t_r cannot be considered as a bound theoretically. But it can still be a good estimate for the maximum noise in case of a ramp input [3.12]. The second issue with the first moment noise metric is that it is limited to ramp inputs. In case of a saturated ramp input, the first moment metric can become significantly pessimistic as illustrated in Figure 3.14.

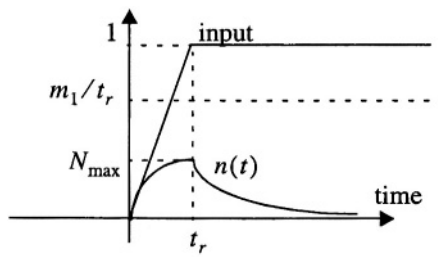


FIGURE 3.14 Noise waveform in case of saturated ramp input

3.5.3 S2P Noise Metric

In case of a saturated ramp input with a rise time t_r, the noise waveform $n(t)$ is the saturated ramp response given in (2.14):

$$n(t) = \begin{cases} \frac{1}{t_r}\left(\sum_i^n \frac{k_i}{p_i^2}e^{p_i t} - \sum_i^n \frac{k_i}{p_i^2} - \sum_i^n \frac{k_i}{p_i}t\right) & 0 \le t < t_r \\ \frac{1}{t_r}\left(\sum_i^n \frac{k_i}{p_i}t_r + \sum_i^n \frac{k_i}{p_i^2}e^{p_i t}(1 - e^{-p_i t_r})\right) & t \ge t_r \end{cases} \tag{3.104}$$

Inserting (3.97) into (3.104) and since $m_0 = 0$ we obtain

$$n(t) = \begin{cases} \frac{1}{t_r}\left(m_1 + \sum_i^n \frac{k_i}{p_i^2}e^{p_i t}\right) & 0 \le t < t_r \\ \frac{1}{t_r}\left(\sum_i^n \frac{k_i}{p_i^2}e^{p_i(t-t_r)}(e^{p_i t_r} - 1)\right) & t \ge t_r \end{cases} \tag{3.105}$$

The above expression requires the poles and residues of the transfer function $H(s)$. A simple and efficient approach is to use the S2P method, which is presented in Section 3.3.4. Its slightly modified version for noise analysis can be outlined as follows:

1. Compute two poles, guaranteed to be stable, from the driving point moments at the input of the aggressor net.
2. At the victim output node, compute the residues according to (3.86). Since there is no dc coupling between the victim and the aggressor nets, that is, $m_0 = 0$, further simplification is possible:

$$k_1 = \frac{m_1 p_2 p_1^2}{p_1 - p_2} \qquad k_2 = \frac{m_1 p_1 p_2^2}{p_2 - p_1} \tag{3.106}$$

The two-pole noise waveform approximation then becomes

$$n(t) = \begin{cases} \dfrac{m_1}{t_r}\left(1 + \dfrac{(p_2 e^{p_1 t} - p_1 e^{p_2 t})}{p_1 - p_2}\right) & 0 \le t < t_r \\ \dfrac{m_1}{t_r(p_1 - p_2)}(p_2(e^{p_1 t_r} - 1)e^{p_1(t - t_r)} - p_1(e^{p_2 t_r} - 1)e^{p_2(t - t_r)}) & t \ge t_r \end{cases} \quad (3.107)$$

Solving for the time point where the derivative of $n(t)$ is equal to zero, we obtain $t_{\max}$ at which maximum coupling occurs:

$$t_{\max} = t_r + \frac{\ln\left(\dfrac{e^{p_2 t_r} - 1}{e^{p_1 t_r} - 1}\right)}{p_1 - p_2} \quad (3.108)$$

It can be easily shown that the second term in (3.108) is positive. Thus $t_{\max}$ is always bigger than t_r. Evaluating the noise formulation at this time point, we obtain the S2P noise metric

$$N_{\max} = \frac{m_1}{t_r(p_1 - p_2)}(p_2(e^{p_1 t_r} - 1)e^{p_1(t_{\max} - t_r)} - p_1(e^{p_2 t_r} - 1)e^{p_2(t_{\max} - t_r)}) \quad (3.109)$$

3.6 RLC Interconnect

To this point we have primarily focused on RC trees and described delay metrics based on modeling the impulse responses by p.d.f.'s. However, if the interconnect tree response is underdamped because of inductance, it can no longer be modeled as a p.d.f., and the aforementioned metrics do not apply [3.13]. A key aspect of RLC delay estimation is first controlling the damping, then approximating the delay. Interestingly, central moments of the impulse response can be used to detect and ultimately guide the controlling of the response underdamping.

In this section we derive the connections between central moments and response damping for a single RLC transmission line. The results are, however, equally applicable to general RLC trees.

3.6.1 RLC Interconnect Delay and Ringing

Because excessive settling time increases delay in some sense, both under-damping and over-damping adversely impact delay. This is evidenced by the responses in Figure 3.15(b) for the series terminated RLC line in Figure 3.15(a). Resistances 15Ω and 25Ω leave the line underdamped which results in a faster but underdamped signal that takes a longer time to settle, while 65Ω results in overdamping.

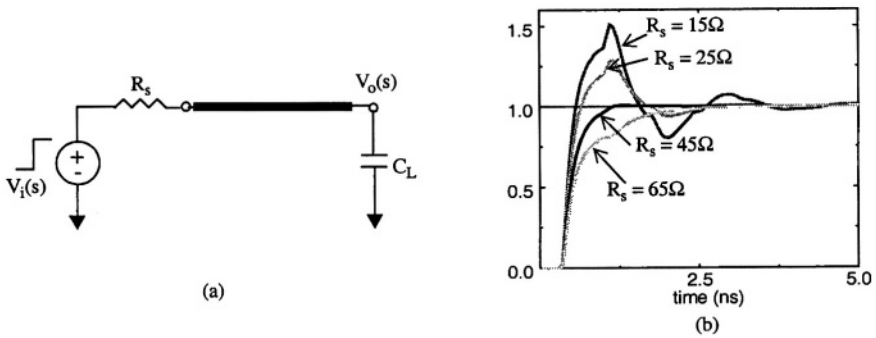


FIGURE 3.15 (a) A source-terminated transmission line. (b) Trade-off between rise-time and ringing for a simple lossy transmission line with capacitive load of 3pF. R=2Ω/cm, L=3.35nH/cm, C=1.34pF/cm, Z_0=50Ω and length=5cm. R_s is the series termination resistance.

3.6.2 Central Moments and Transmission Line Response

For the simple source-terminated transmission line shown in Figure 3.15(a), the transfer function is

$$H(s) = \frac{V_o(s)}{V_i(s)} = \frac{1}{sC_L(R_s\cosh(\gamma d) + Z_0\sinh(\gamma d)) + (R_s/Z_0)\sinh(\gamma d) + \cosh(\gamma d)} \tag{3.110}$$

where $\gamma = \sqrt{(R+sL)(sC)}$ is the *propagation function* and $Z_0 = \sqrt{(R+sL)/(sC)}$ is the *characteristic impedance* of the line. *R, L* and *C* are the per-unit-length resistance, inductance, and capacitance parameters of the transmission line, respectively, *d*

is the length of the line, and the series resistance is given by $R_s = R_{dr} + R_{ter}$, where R_{dr} is the driver resistance and R_{ter} the additional termination resistance. The dielectric loss, G, is assumed to be negligibly small. From (3.110), the moments of the transfer function of the transmission line system can be obtained as a function of R_s, or as a function of the series termination resistance, R_{ter}. The driver resistance R_{dr} is assumed to be linear.

A. Lossless Interconnect

For an unloaded lossless transmission line driven by a step input, as shown in Figure 3.16(a), it is well known that the optimal termination resistance is $R_s = Z_0$ [3.14]. With this termination, the ideal signal is the input step delayed by the time-of-flight along the line, given by $T_f = \sqrt{LC}d$. Thus, the ideal response, $\tilde{v}_o(t)$, is as shown in Figure 3.16(b). The following discussion shows that this ideal response is indeed obtained when the central moments of the impulse response are minimized.

For the lossless line in Figure 3.16(a), the transfer function is given by

$$H(s) = \frac{1}{(R_s/Z_0)\sinh(\gamma d) + \cosh(\gamma d)} \tag{3.111}$$

where $\gamma = s\sqrt{LC}$ and $Z_0 = \sqrt{L/C}$. For this transfer function, the second and third central moments of the impulse response are symbolically given as:

$$\begin{aligned} \mu_2 &= -CLd^2 + R_s^2C^2d^2 \\ \mu_3 &= -2R_sC^2Ld^3 + 2C^3d^3R_s^3 \end{aligned} \tag{3.112}$$

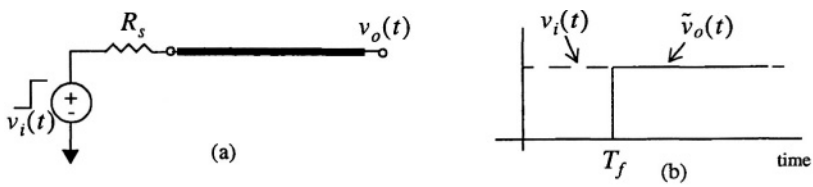


FIGURE 3.16 (a) A perfect lossless, unloaded transmission line. Time-of-flight, $T_f = \sqrt{LC}d$, where d is the length of the line. (b) Input $v_i(t)$ and ideal output response $\tilde{v}_o(t)$.

Symbolically solving $\mu_2 = 0$ yields two roots:

$$\sqrt{\frac{L}{C}}, \qquad -\sqrt{\frac{L}{C}} \tag{3.113}$$

Solving $\mu_3 = 0$ for R_s yields three roots:

$$0, \qquad \sqrt{\frac{L}{C}}, \qquad -\sqrt{\frac{L}{C}} \tag{3.114}$$

The positive root provides the solution, $R_s = Z_0$. Then, the transfer function given in (3.111) becomes

$$H(s) = \frac{1}{\sinh(\gamma d) + \cosh(\gamma d)} = e^{-sT_f} \tag{3.115}$$

where $T_f = \sqrt{LC}d$ is the time-of-flight. Then it is easy to show that this transfer function provides the desired ideal waveform at the output of the transmission line

$$v_o(t) = v_i(t - T_f) \tag{3.116}$$

From above, it can be inferred that the ideal impulse response for a lossless transmission line is symmetric and localized (zero dispersion) about its mean, $\mu = \sqrt{LC}d$. Conversely, forcing the impulse response to be symmetric and localized about the mean ensures critical damping.

B. Lossy Transmission Lines

Lossy lines, on the other hand, display the phenomenon of "dispersion" which is due to the dependence of the phase velocity of a propagating wave on the frequency of the wave [3.14]. Consequently, for a signal comprised of a band of frequencies, the different frequency components do not maintain the same phase relationships as they propagate down the line. Due to this inherent property of dispersion of a lossy transmission line, μ_2 (being a measure of dispersion) cannot vanish for a lossy line. Thus, our objective for terminating a lossy transmission line is to solve for $\mu_3 = 0$ with minimum μ_2.

For a positive function $h(t)$, since the third central moment μ_3 is known to be a measure of the *asymmetry* of the function, $\mu_3 > 0$ represents a positively skewed function

with a long right tail, as shown in Figure 3.17(b), and corresponds to an overdamped signal (Figure 3.17(a)). For a transmission line system, when the response is underdamped, there is some overshoot/undershoot, and so, for the impulse response shown in Figure 3.17(b), $\mu_3 < 0$. The third central moment from equation (2.57) can be rewritten as

$$\mu_3 = \underbrace{\int_0^{\mu} (t-\mu)^3 h(t)dt}_{\mu_3^-} + \underbrace{\int_{\mu}^{\infty} (t-\mu)^3 h(t)dt}_{\mu_3^+} \tag{3.117}$$

so that for a symmetric $h(t) \geq 0$, $\mu_3^- = -\mu_3^+ \Rightarrow \mu_3 = 0$, which corresponds to a critically damped system.

Thus, for a lossy transmission line, the objective is to design the impulse response to be localized and symmetric about its mean. And since μ_3 is a measure of asymmetry, and hence ringing, $\mu_3 = 0$ with minimum μ_2 is proposed to be the condition for optimal termination for a lossy transmission line. However, in order to minimize ringing while maximizing the signal slope, trade-offs between ringing versus rise-time should be considered using the second central moment, μ_2 [3.13].

Finally, it should be noted that if the RLC line damping is controlled by forcing $\mu_3 = 0$, the distribution is more symmetric than an RC impulse response, and, therefore, the Elmore delay metric is more accurate since the mean is closer to the median.

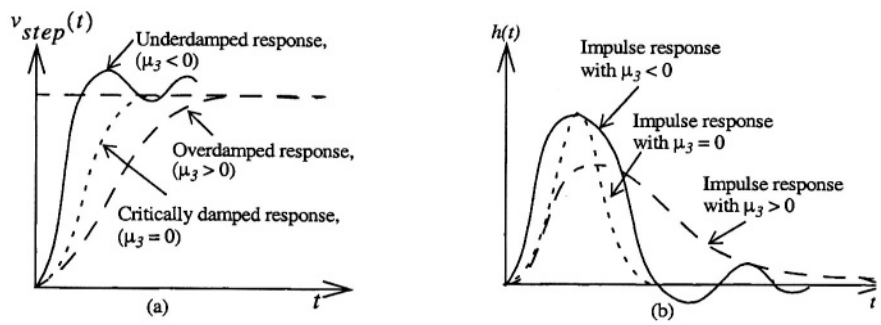


FIGURE 3.17 (a) Underdamped, critically damped, and overdamped step responses. (b) Impulse responses with positive, zero, and negative μ_3's corresponding to overdamping, critical damping, and underdamping, respectively.

3.7 Signal Attenuation and Phase Delay in RLC Trees

Until now we have focused on transient delay calculation in RLC interconnect, but there are occasions -- e.g. clock signals such as the one shown in Figure 3.18 -- for which steady-state characterization is more effective. If the fundamental frequency of the periodic signal in Figure 3.18 is f, then we know that all other signal content is at $2f$, $3f$, $4f$, etc., along with a dc signal component at $f = 0$. But even for the nearly ideal input signal shown in Figure 3.18, there is very little signal energy at frequency $2f$, and even less at $3f$, $4f$, and so on. In addition, due to the low-pass nature of the interconnect, we expect the ratio of the signal energy at $2f$ to that at f to be even less at the output nodes (latch points). Furthermore, with frequencies passing 1 GHz, clock signals appear more like sine waves. For this reason we focus the clock tree (or mesh) design on balancing the phase delays (see Figure 3.18) and controlling the attenuation of the output frequency response at the fundamental frequency f.

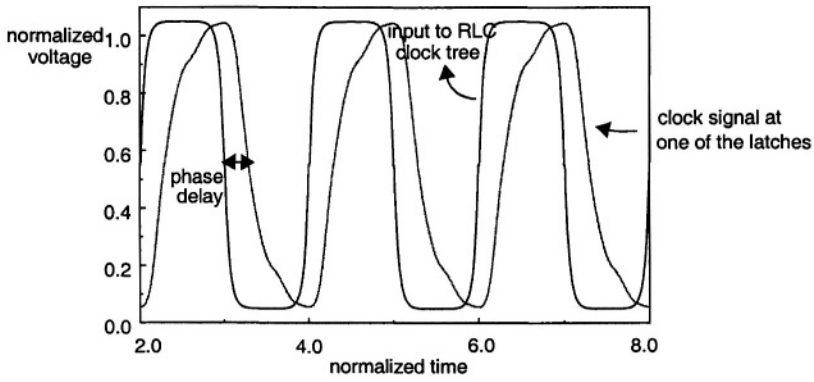


FIGURE 3.18 A typical input-output waveform pair for a commercial RLC clock tree [3.15].

While it is desirable to measure the clock distribution performance in the frequency domain, a complete ac analysis is too costly in terms of runtime for most optimization and synthesis tools. In this section, we derive simple, yet accurate metrics for the phase delay and signal attenuation of RLC clock trees. The attenuation metric also provides an estimation of the damping of RLC circuit responses. When the response is overdamped, we prove that the phase delay is bounded by the first moment of the impulse response, which is the ubiquitous Elmore delay. Importantly, this bound is extremely accurate over the range of operating frequency for which the attenuation

level is acceptable. The attenuation metric is based on the first and second moments of the impulse response and also provides a tight upper bound near the operating frequencies of interest. For underdamped RLC responses the first moment metric is not a bound, but the attenuation metric correctly indicates that the design is not a reliable one anyway. When the circuit is underdamped, or there are significant signal reflections, the attenuation metric will predict this by specifying a gain greater than unity.

3.7.1 Attenuation Metric for RLC Trees

Consider the moment expansion of a general transfer function,

$$H(s) = m_0 + m_1 s + m_2 s^2 + \ldots \tag{3.118}$$

and let $F(s)$ be the magnitude squared function in the Laplace domain:

$$F(s) = H(s)H(-s) \tag{3.119}$$

The Taylor series expansion of $F(s)$ about $s = 0$ can be written as

$$F(s) = a_0 + a_2 s^2 + a_4 s^4 + \ldots \tag{3.120}$$

The first few coefficients in (3.120) can be expressed in terms of the moments as follows:

$$\begin{aligned} a_0 &= m_0^2 \\ a_2 &= 2m_0 m_2 - m_1^2 \\ a_4 &= 2m_0 m_4 - 2m_1 m_3 + m_2^2 \\ &\ldots \end{aligned} \tag{3.121}$$

The amplitude function, $A(\omega)$, which is defined in (2.16), is related to $F(s)$ as

$$A^2(\omega) = F(s)\big|_{s = j\omega} \tag{3.122}$$

Therefore, the square of the amplitude function of a linear circuit can be expressed in terms of moments as

$$A^2(\omega) = m_0^2 - \omega^2(2m_2 - m_1^2) + \omega^4(2m_4 - 2m_1m_3 + m_2^2) - \omega^6(\ldots) + \ldots \quad (3.123)$$

Note that the above expansion is true for general linear circuits. But for RC tree responses, we can identify the conditions for which it is acceptable to approximate the amplitude function by a limited number of series expansion terms.

Assume that p_1 is the smallest modulus pole for an RC tree transfer function, $H(s)$. In Subsection 3.3.3, we showed that the ratio of two successive moments approaches the first pole as the order increases, i.e.,

$$\lim_{i \to \infty} \frac{m_i}{m_{i+1}} = p_1 \quad (3.124)$$

The above relation implies that the moments are approximately scaled by the inverse of the smallest modulus pole when the order is increased by one, and in reality it is true even for very low order moments. Therefore, we can approximate the higher order moments in terms of the zeroth order moment and the smallest modulus pole as

$$m_i \approx \frac{m_0}{p_1^i}, \qquad i = 1, 2, \ldots \quad (3.125)$$

Since $m_0 = 1$ for tree circuits, (3.125) can also be expressed as

$$m_i = \frac{\alpha_i}{p_1^i}, \qquad i = 0, 1, \ldots \quad (3.126)$$

where

$$\alpha_i \approx 1. \quad (3.127)$$

For example, consider the clock tree circuit mentioned in Section 2.8. For the response at the output node, the first seven values of α_i are: 1.000, 1.020, 1.013, 1.006, 0.999, 0.994, 0.988. Note that the accuracy of the above approximation is not important because it is not directly used in the metric derivations. Only the order of α_i's are required.

Substituting (3.126) into (3.123) and assuming that $m_0 = 1$, we obtain

$$A^2(\omega) = 1 - \left(\frac{\omega}{p_1}\right)^2 (2\alpha_2 - \alpha_1^2) + \left(\frac{\omega}{p_1}\right)^4 (2\alpha_4 - 2\alpha_1\alpha_3 + \alpha_2^2) - \left(\frac{\omega}{p_1}\right)^6 (\ldots) + \ldots \quad (3.128)$$

It follows from (3.128) that $A^2(\omega)$ is a polynomial in (ω/p_1) having coefficients on the order of unity. Since for reliable circuit operation the attenuation must be low[1], the fundamental frequency of the clock signal, ω, should be less than the first cut-off frequency, $\omega_c = -p_1$. Therefore, the higher order terms in (3.128) decrease very quickly, and the amplitude response can be approximated as

$$A^2(\omega) \approx 1 - \omega^2(2m_2 - m_1^2) \quad (3.129)$$

Since the second central moment is $\mu_2 = 2m_2 - m_1^2$, the above metric becomes

$$A^2(\omega) \approx 1 - \omega^2\mu_2 \quad (3.130)$$

In Appendix 3.A.3, it is shown that for RC trees, the coefficients in (3.120) are positive, therefore the series terms in (3.123) alternate in sign. And since it is a power series with approximately unity coefficients, the error is easily bounded by the first truncated term. Therefore, we conclude that for RC trees, the approximation given in (3.130) is always a lower bound for the amplitude response and hence an upper bound for the attenuation. This is demonstrated in Fig. 3.19 for an RC clock tree example.

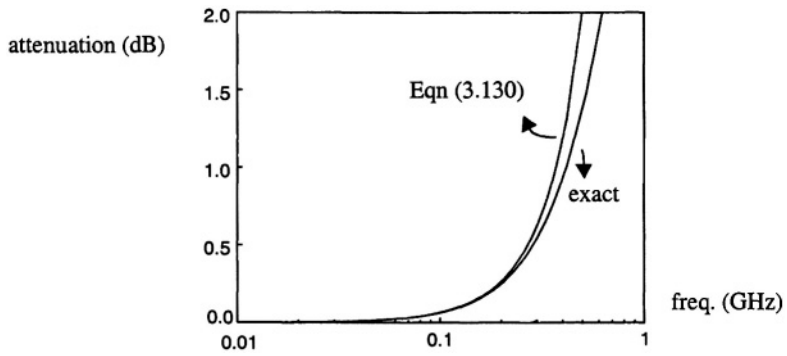


FIGURE 3.19 The attenuation and its approximation for an RC clock tree.

1. Signal attenuation of more than 10% is generally unacceptable for aggressive designs due to the tight noise margins. This is particularly the case for scaled supply voltage designs, where the threshold voltages, hence gate switching points, do not directly scale with the supply voltages.

The proof in Appendix 3.A.3 that the coefficients in (3.120) are positive relies on the fact that the RC tree impulse response is always positive. In Section 3.6, it is shown that the second central moment, μ_2, is negative only if the RLC circuit is underdamped. Therefore, for critically and overdamped RLC trees, in which the impulse responses are always positive, the alpha-term coefficients in (3.128) are positive and the first two terms can be used as a lower bound for the amplitude response:

$$A^2(\omega) \approx 1 - \omega^2 \mu_2 \tag{3.131}$$

Note that it is apparent from (3.131) that a negative μ_2 corresponds to an underdamped response.

As an example we consider the response of an RLC clock tree (344 inductors, 658 capacitors, and 1267 resistors) that was extracted from a commercial microprocessor [3.15]. The amplitude responses for two different values of driver resistance are shown in Figure 3.20. Included are the amplitude approximation from (3.131) for each case. In both cases the approximation behaves as expected: clearly an upper bound for the overdamped case, and clearly indicates underdamping for the oversized driver case. The corresponding step responses are shown in Figure 3.21. Clearly, the underdamped response amplitude and phase delay are not of interest since the periodic clock signal in such a case would be unacceptable.

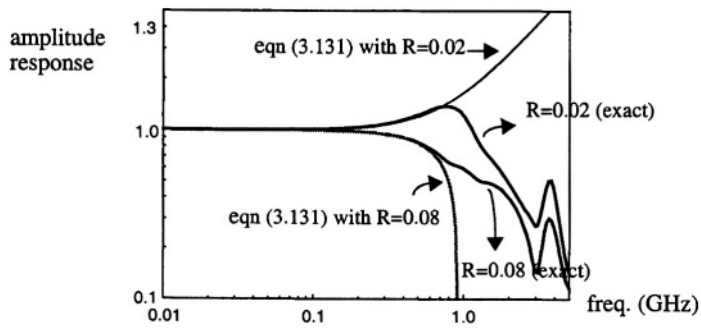


FIGURE 3.20 Amplitude response of an RLC clock tree for two different source resistances, R= 0.08 (overdamped), and R=0.02 (underdamped).

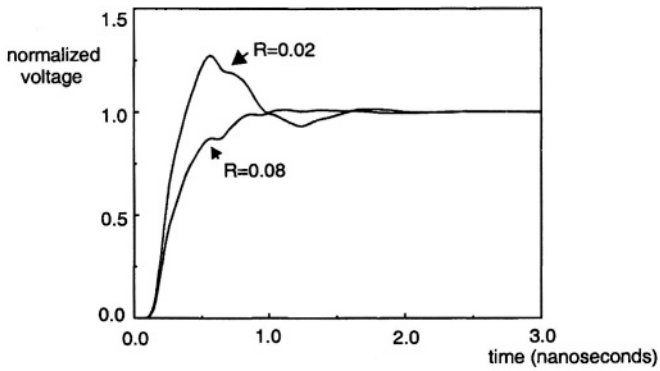


FIGURE 3.21 Step response of the RLC clock tree.

3.7.2 Phase Delay Metric for RLC Trees

From (3.118) we can write

$$\begin{aligned} H(j\omega) &= m_0 + m_1(j\omega) + m_2(j\omega)^2 + \ldots \\ &= (m_0 - m_2\omega^2 + m_4\omega^4 - \ldots) + j(m_1\omega - m_3\omega^3 + m_5\omega^5 - \ldots) \end{aligned} \tag{3.132}$$

From which, we can express the phase function as

$$\theta(\omega) = \operatorname{atan}(f(\omega)) \tag{3.133}$$

where

$$f(\omega) = \frac{m_1\omega - m_3\omega^3 + m_5\omega^5 - \ldots}{m_0 - m_2\omega^2 + m_4\omega^4 - \ldots} \tag{3.134}$$

The Taylor series expansion of the phase function in terms of $f(\omega)$ is given by

$$\theta(\omega) = f(\omega) - \frac{[f(\omega)]^3}{3} + \frac{[f(\omega)]^5}{5} - \ldots \tag{3.135}$$

Similarly the Taylor series expansion of $f(\omega)$ is

$$f(\omega) = b_1\omega + b_3\omega^3 + b_5\omega^5 + \ldots \tag{3.136}$$

Assuming $m_0 = 1$, the first few terms of (3.136) are calculated as

$$\begin{aligned} b_1 &= m_1 \\ b_3 &= -m_3 + m_1 m_2 \\ b_5 &= m_5 - m_1 m_4 - m_2 m_3 + m_1 m_2^2 \\ &\ldots \end{aligned} \tag{3.137}$$

Substituting (3.136) into (3.135) we obtain the power series expansion for $\theta(\omega)$:

$$\theta(\omega) = b_1\omega + \left(b_3 - \frac{b_1^3}{3}\right)\omega^3 + \left(b_5 - b_1^2 b_3 + \frac{b_1^5}{5}\right)\omega^5 + \ldots \tag{3.138}$$

Substituting first (3.137) and then (3.126) into (3.138) yields

$$\begin{aligned} \theta(\omega) = {} & \frac{m_1\omega}{p_1} + \left(-\alpha_3 + \alpha_1\alpha_2 - \frac{\alpha_1^3}{3}\right)\left(\frac{\omega}{p_1}\right)^3 \\ & + \left(\alpha_5 - \alpha_1\alpha_4 + \alpha_1\alpha_2^2 - \alpha_2\alpha_3 - \alpha_1^3\alpha_2 + \alpha_1^2\alpha_3 + \frac{\alpha_1^5}{5}\right)\left(\frac{\omega}{p_1}\right)^5 + \ldots \end{aligned} \tag{3.139}$$

Similar to (3.128), the phase function is a polynomial in (ω/p_1) having coefficients on the order of unity.

Therefore, using (2.18), the phase delay can be approximated as

$$\tau(\omega) = -\frac{\theta(\omega)}{\omega} \approx -m_1 - \left(-m_3 + m_1 m_2 - \frac{m_1^3}{3}\right)\omega^2 \tag{3.140}$$

From the definition of the third central moment in (2.57) it follows that

$$\tau(\omega) \approx T_D - \frac{\mu_3\omega^2}{6} \tag{3.141}$$

Since the third central moment, μ_3, is always positive for overdamped RLC tree responses, equation (3.141) also indicates that the Elmore delay is an upper bound for the phase delay as we previously proved in Section 2.8.

Considering once again the RLC clock tree example used in Figure 3.20 and Figure 3.21, the phase delays are compared with the Elmore delays for two different source resistances in Fig. 3.22. Note that (3.141) could be used to extend the range of validity for the phase delay metric, however, a comparison of Figure 3.20 and Figure 3.22 shows that the Elmore delay begins to become inaccurate only when the attenuation begins to become unsupportable (greater than 10%). For this reason we would conclude that it is unnecessary to use a more accurate delay metric, since the Elmore delay provides excellent accuracy for the frequency range of acceptable attenuation.

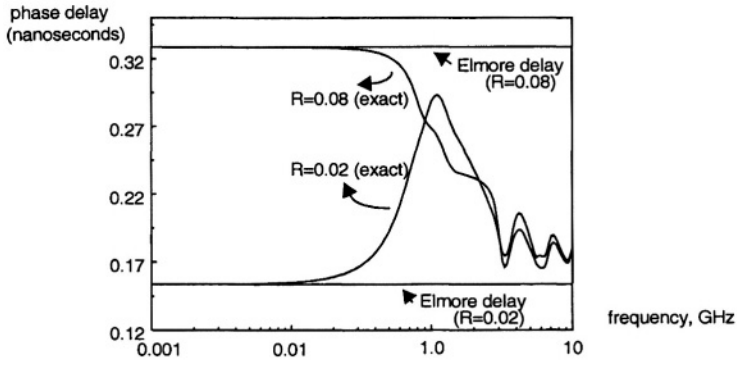


FIGURE 3.22 Comparison of the phase delay with the Elmore delay for two different sources resistance values in the RLC tree.

This observation has two important implications. First, the widely used Elmore delay remains an acceptable metric for RLC clock tree synthesis. Secondly, but perhaps most importantly, it is critical for the clock tree synthesis algorithm to focus on controlling the signal attenuation, even more than the delay/skew initially.

3.8 Summary

Inspired by Elmore, a myriad of moment based delay metrics have evolved based on higher order moments and extensions of the analogy between p.d.f.'s and RC impulse responses. In addition, other interesting properties of moments were shown to facilitate approximations of RLC interconnect damping control, cross-talk estimation, and steady-state phase delay prediction. While these metrics extend the utility of moments for performance prediction, like the Elmore delay they all have their limitations and must be used judiciously.

In the next several chapters we explore more general algorithms for modeling interconnect more precisely via moments and related subspaces.

3.A Appendix

3.A.1 Moments of RC Tree Admittance Functions

Let $Y(s)$ be a driving point admittance function in an RC tree and consider its representation in terms of poles and residues

$$Y(s) = k_0 + \sum_{j=1}^{q} \frac{k_j}{s - p_j} \tag{3.142}$$

Assume a moment expansion for $Y(s)$ in the form

$$Y(s) = y_0 + y_1 s + y_2 s^2 + \ldots \tag{3.143}$$

In terms of the poles and residues, the moments are given by

$$y_i = -\sum_{j=1}^{q} \frac{k_j}{p_j^{i+1}}, \qquad i > 0 \tag{3.144}$$

Let us define

$$f = y_{i-1} \cdot y_{i+1} - y_i \cdot y_i \tag{3.145}$$

Then we have

$$\begin{aligned}
f &= \sum_n \frac{k_n}{p_n^i} \cdot \sum_n \frac{k_n}{p_n^{i+2}} - \sum_n \frac{k_n}{p_n^{i+1}} \cdot \sum_n \frac{k_n}{p_n^{i+1}} \\
&= \sum_n \frac{k_n^2}{p_n^{2i+2}} + \sum_n \sum_{l \neq n} \frac{k_n k_l}{p_n^i \cdot p_l^{i+2}} - \sum_n \frac{k_n^2}{p_n^{2i+2}} - \sum_n \sum_{l \neq n} \frac{k_n k_l}{p_n^{i+1} \cdot p_l^{i+1}} \\
&= \sum_n \sum_{l \neq n} \frac{k_n k_l}{p_n^i \cdot p_l^{i+2}} - \sum_n \sum_{l \neq n} \frac{k_n k_l}{p_n^{i+1} \cdot p_l^{i+1}} \\
&= \sum_n \sum_{l=n+1} k_n k_l \left(\frac{1}{p_n^i \cdot p_l^{i+2}} + \frac{1}{p_n^{i+2} \cdot p_l^i} \right) - \sum_n \sum_{l=n+1} 2 k_n k_l \frac{1}{p_n^{i+1} \cdot p_l^{i+1}} \\
&= \sum_n \sum_{l=n+1} \frac{k_n k_l}{p_n^{i+2} p_l^{i+2}} (p_n^2 + p_l^2 - 2 p_n p_l) \\
&= \sum_n \sum_{l=n+1} \frac{k_n k_l}{p_n^{i+2} p_l^{i+2}} (p_n - p_l)^2
\end{aligned} \tag{3.146}$$

Since the residues are negative and repetitive poles are impossible for RC driving point functions [3.16], we have $f > 0$. Therefore from (3.145), we have

$$y_{i-1} \cdot y_{i+1} > y_i \cdot y_i \tag{3.147}$$

Since either y_i or y_{i+1} is positive and the other one is negative, we obtain

$$\frac{y_{i-1}}{y_i} < \frac{y_i}{y_{i+1}} \tag{3.148}$$

3.A.2 Realness of the S2P Poles

We prove that the roots of $Q(s)$ in (3.55) are real numbers.

In order to have real roots, the coefficients of $Q(s)$ have to satisfy

$$a_1^2 - 4a_2 = \frac{\left(\frac{y_{i+3}}{y_{i+2}}\right)^2\left(\frac{y_{i+2}}{y_{i+3}} - \frac{y_i}{y_{i+1}}\right)^2 - 4\left(\frac{y_{i+3}}{y_{i+1}}\right)\left(\frac{y_{i+2}}{y_{i+3}} - \frac{y_{i+1}}{y_{i+2}}\right)\left(\frac{y_{i+1}}{y_{i+2}} - \frac{y_i}{y_{i+1}}\right)}{\left(\frac{y_{i+1}}{y_{i+2}} - \frac{y_i}{y_{i+1}}\right)^2} > 0 \quad (3.149)$$

where a_1 and a_2 are defined in (3.80). Since the denominator is positive, it is sufficient to show that the numerator in (3.149) is always positive in order to have real coefficients. Let us define

$$x = \left(\frac{y_{i+2}}{y_{i+3}} - \frac{y_{i+1}}{y_{i+2}}\right) \qquad y = \left(\frac{y_{i+1}}{y_{i+2}} - \frac{y_i}{y_{i+1}}\right) \qquad z = x + y = \left(\frac{y_{i+2}}{y_{i+3}} - \frac{y_i}{y_{i+1}}\right) \quad (3.150)$$

It follows from (3.82) that

$$x > 0 \qquad y > 0 \quad (3.151)$$

Substituting (3.150) into (3.149), the numerator of (3.149) can be written as

$$D = \left(\frac{y_{i+3}}{y_{i+2}}\right)^2 z^2 - 4\left(\frac{y_{i+3}}{y_{i+1}}\right)xy \quad (3.152)$$

Substituting $z = x + y$ in (3.152) and defining $W = D(y_{i+2}/y_{i+3})^2$, we get

$$W = x^2 + y^2 + 2xy - 4xy\left(\frac{y_{i+2}^2}{y_{i+1}y_{i+3}}\right) \quad (3.153)$$

Using (3.82), it can be shown that $y_{i+1}y_{i+3} > y_{i+2}^2$. Thus

$$0 < \frac{y_{i+2}^2}{y_{i+1}y_{i+3}} < 1 \quad (3.154)$$

Combining (3.154) and (3.153), we obtain

$$W > (x - y)^2 > 0 \quad (3.155)$$

hence $D > 0$, which completes our proof.

3.A.3 Moments of Magnitude Squared Functions

Let $H(s)$ be a transfer function in an RC tree and consider its magnitude squared function, $F(s)$,

$$F(s) = H(s)H(-s) \tag{3.156}$$

On the $j\omega$ axis, $F(s)$ becomes the square of the magnitude response. Let $f(t)$ be the inverse Laplace transform of $F(s)$. From (3.156) it follows that

$$f(t) = h(t)*h(-t) \tag{3.157}$$

where $h(t)$ is the impulse response of the circuit and * is the convolution operator. In the previous chapter, it was shown that the impulse response of an RC tree is always positive. Consequently, $f(t)$ is also always positive and it is an even function of t. Next consider the Taylor series expansion of $F(s)$,

$$\begin{aligned} F(s) &= \int_{-\infty}^{\infty} f(t)e^{-st}dt \\ &= \sum \frac{(-1)^k}{k!}s^k \int_{-\infty}^{\infty} t^k f(t)dt \end{aligned} \tag{3.158}$$

Since $f(t)$ is an even function of t and always positive, the integral in the summation is zero for odd order terms and a positive number for even order terms. Therefore, $F(s)$ can be written as

$$F(s) = f_0 + f_2 s^2 + f_4 s^4 + \ldots \tag{3.159}$$

where all the coefficients are positive.

References

[3.1] R. Kay and L. Pileggi, "PRIMO: Probability Interpretation of Moments for Delay Calculation," in *Proceedings of the Design Automation Conference,* pp. 463-468, Jun. 1998.

[3.2] M. H. DeGroot, *Probability and Statistics,* Second Edition. Reading, MA: Addison Wesley, 1989.

[3.3] T. Lin, E. Acar, and L. T. Pileggi, "h-gamma: an RC delay metric based on a gamma distribution approximation of the homogeneous response," in *Tech. Dig. 1998 ACM/IEEE Int. Conf. on Computer-Aided Design,* pp. 19-24, Nov. 1998.

[3.4] J. Rubinstein, P. Penfield, Jr., and M.A. Horowitz, "Signal delay in RC tree networks," *IEEE Trans. on Computer Aided Design,* vol. CAD-2, pp. 202-211, 1983.

[3.5] M. A. Horowitz, "Timing models for MOS circuits," Ph.D. thesis, Stanford University, 1980.

[3.6] A. Odabasioglu, M. Celik, and L. T. Pileggi, "Efficient and accurate delay metrics for RC interconnect," *PATMOS: Int. Workshop on Power and Timing Modeling, Optimization, and Simulation,* Oct. 1999.

[3.7] B. Tutuianu, F. Dartu, and L. Pileggi, "An explicit RC-circuit delay approximation based on the first three moments of the impulse response", in *Proc. ACM/IEEE Design Automation Conf.,* 1996

[3.8] E. Acar, A. Odabasioglu, M. Celik, and L. T. Pileggi, "S2P: A stable 2-pole RC delay and coupling noise metric," in *Proc. of Great Lakes Symposium on VLSI,* 1999.

[3.9] A. B. Kahng and S. Muddu, "An analytical delay model for RLC interconnects," *IEEE Trans. on Computer-Aided Design,* vol. 16, no. 12, pp. 1507-1514,1997.

[3.10] A. B. Kahng and S. Muddu, "Accurate analytical delay models for VLSI interconnects," Univ. of California, Los Angeles, UCLA CS Dept., TR-950034, Sept. 1995.

[3.11] C. J. Alpert, A. Devgan, and C. V. Kashyap, "RC delay metrics for performance optimization," *IEEE Trans. on Computer-Aided Design,* vol. 20, no. 5, pp. 571-582, May 2001.

[3.12] A. Devgan, "Efficient coupled noise estimation for on-chip interconnects," in *Tech. Dig. 1997 IEEE/ACM Int. Conf. on Computer-Aided Design,* pp 147-151, Nov. 1997.

[3.13] R. Gupta, B. Krauter, and L. T. Pileggi, "Transmission line synthesis via constrained multivariable optimization," *IEEE Trans. on Computer-Aided Design,* vol. 16, no. 1, pp. 6-19, Jan. 1997.

[3.14] N. N. Rao, *Elements of Engineering Electromagnetics.* Englewood Cliffs, NJ: Prentice-Hall, 1991.

[3.15] M. Celik and L. Pileggi, "Metrics and bounds for phase delay and signal attenuation in RC(L) clock trees," *IEEE Trans. on Computer Aided Design,* vol. CAD-18, Mar. 1999.

[3.16] M.E. Van Valkenburg, *Introduction to Modern Network Synthesis.* John Wiley & Sons Inc., 1960.

CHAPTER 4

Asymptotic Waveform Evaluation

In the previous chapter, we saw that moment matching can be used to fit circuit responses to probability distribution functions to construct simple and accurate delay metrics. Alternatively, moment matching can be applied to varying orders of circuit-response functions by using moments to calculate approximate poles and zeros. This chapter describes the asymptotic waveform evaluation (AWE) technique which efficiently calculates circuit moments via recursive solution of a dc equivalent circuit, then employs a moment matching method to compute Padé approximations of circuit functions. We begin where we left off with the description of moments in Chapter 2, then outline general moment matching and its accuracy and stability properties.

4.1 State Equations and Circuit-Response Functions

Asymptotic waveform evaluation can best be explained via state-space formulation. Consider a lumped, linear, time-invariant circuit which has one input and one output. The standard form of the state equations of such a circuit is

$$\frac{d\mathbf{x}}{dt} = \mathbf{A}\mathbf{x} + \mathbf{b}u$$
$$y = \mathbf{c}^T\mathbf{x} + du \tag{4.1}$$

where $\mathbf{x}$ is an n-dimensional column vector (the state vector), $\mathbf{A}$ is an $n \times n$ matrix (the state matrix), $\mathbf{b}$ and $\mathbf{c}$ are n-dimensional column vectors, u is the input variable, y is the output variable, and d is a scalar. Since the direct coupling term, d, can be computed separately, for simplicity, we will assume $d = 0$.

The zero-state impulse response of the linear circuit defined in (4.1) is defined as

$$h(t) = y(t)\big|_{u(t)=\delta(t)} \tag{4.2}$$

of which the Laplace transform $L\{h(t)\}$ is the *circuit function* $H(s)$.

Upon applying the Laplace transform to state equations, we have

$$s\mathbf{X}(s) - \mathbf{x}(0) = \mathbf{A}\mathbf{X}(s) + \mathbf{b}U(s) . \tag{4.3}$$

Neglecting the initial conditions, that is, with $\mathbf{x}(0) = \mathbf{0}$, we obtain

$$\mathbf{X}(s) = (s\mathbf{I} - \mathbf{A})^{-1}\mathbf{b}U(s) . \tag{4.4}$$

Thus the circuit function $H(s)$ is given by

$$H(s) = \mathbf{c}^T(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{b} \tag{4.5}$$

which can also be expressed in one of the following forms:

$$\begin{aligned} H(s) &= \frac{B(s)}{A(s)} = \frac{b_0 + b_1 s + \ldots + b_{n-1}s^{n-1}}{1 + a_1 s + \ldots + a_n s^n} \\ &= K\frac{(s-z_1)\cdot(s-z_2)\ldots(s-z_{n-1})}{(s-p_1)\cdot(s-p_2)\ldots(s-p_n)} \\ &= \frac{k_1}{s-p_1} + \frac{k_2}{s-p_2} + \ldots + \frac{k_n}{s-p_n} \end{aligned} \tag{4.6}$$

We recognize that the poles, p_i, of $H(s)$ are the roots of the characteristic polynomial

$$A(s) = \det(s\mathbf{I} - \mathbf{A}) \tag{4.7}$$

or the eigenvalues of the matrix $\mathbf{A}$.

Once the circuit function is obtained in one of the forms shown in (4.6), the time- and frequency-domain responses can be computed efficiently as outlined in Chapter 2. For typical interconnect circuits, however, the number of poles can easily be of the order of hundreds or thousands. Obviously, it is impractical to attempt to find all of the circuit poles, especially since some have insignificant contributions to the circuit performance. What we seek instead is an efficient means to obtain those few dominant poles that adequately characterize circuit behavior. One of the most effective procedures for approximating such a set of dominant poles (and zeros) is via moment matching.

Asymptotic waveform evaluation (AWE), first introduced in 1989 by Pileggi and Rohrer [4.1], employs a form of moment matching to approximate the behavior of linear circuits in either the time or the frequency domain. AWE, and moment matching in general, consist of two main stages:

1. Moment generation from the circuit
2. Moment matching

Specifically, AWE finds an approximation for the circuit function $H(s)$ in the form

$$\hat{H}(s) = \frac{\hat{b}_0 + \hat{b}_1 s + \ldots + \hat{b}_{q-1} s^{q-1}}{1 + \hat{a}_1 s + \ldots + \hat{a}_q s^q} \tag{4.8}$$

where the reduced order q is much less than the original order n. AWE achieves this by first generating the moments of $H(s)$ from the circuit, and then matching the first $2q$ moments to the low-order q-pole model given in (4.8).

The rest of this chapter analyzes AWE in detail, beginning with a description of moments in the following section. Efficient ways of generating moments is the subject of the next chapter.

4.2 Moments

When the input is an impulse function, that is, $U(s) = 1$, the state vector $\mathbf{X}(s)$ in (4.4) becomes

$$\mathbf{X}(s) = (s\mathbf{I} - \mathbf{A})^{-1}\mathbf{b} \tag{4.9}$$

and it can be represented with a Taylor series expansion about $s = 0$,

$$\mathbf{X}(s) = \mathbf{x}_0 + \mathbf{x}_1 s + \mathbf{x}_2 s^2 + \ldots \tag{4.10}$$

The coefficient $\mathbf{x}_i$ in (4.10) is called the ith *moment vector* of the circuit. Similarly, the output response can be represented with a Taylor series expansion in the form

$$H(s) = m_0 + m_1 s + m_2 s^2 + \ldots \tag{4.11}$$

where m_i is the ith *moment* of the circuit function. From (4.5), (4.10), and (4.11), it follows that the ith moment of the circuit function is calculated from the moment vector as

$$m_i = \mathbf{c}^T \mathbf{x}_i \tag{4.12}$$

From (4.11) we can show the relation between the circuit function and its moments in the frequency domain:

$$m_i = \frac{\left.\frac{d^i}{ds} H(s)\right|_{s=0}}{i!} \tag{4.13}$$

In the time domain, as we saw in Chapter 2, the moments are related to the impulse response, $h(t)$, as

$$m_i = \frac{(-1)^i}{i!} \int_0^\infty t^i h(t)\, dt \tag{4.14}$$

Having defined the moments and moment vectors, we now turn our attention to moment computation. The state moment vectors in (4.10) can be shown to be

$$\mathbf{x}_j = -\mathbf{A}^{-j-1}\mathbf{b} \tag{4.15}$$

which provides the following recursion for computing the moments:

$$\begin{aligned} \mathbf{x}_0 &= -\mathbf{A}^{-1}\mathbf{b} \\ \mathbf{x}_j &= \mathbf{A}^{-1}\mathbf{x}_{j-1} \qquad j > 0 \end{aligned} \tag{4.16}$$

Once the state vector moments are calculated, the moments of any circuit function can be simply computed using (4.12). To start the recursion in (4.16) we need to compute $\mathbf{x}_0 = -\mathbf{A}^{-1}\mathbf{b}$. This requires LU decomposition of the matrix **A**. The computation for each additional moment, however, is far less costly requiring only forward and back substitutions.

The above recursive relation is useful, but we would rather not formulate the state matrix **A** if we can avoid it. In general, the **A** matrix for a lumped, linear circuit takes the following form [4.2]:

$$\mathbf{A} = \begin{bmatrix} \mathbf{C} & \mathbf{0} \\ \mathbf{0} & \mathbf{L} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{H}_{CC} & \mathbf{H}_{CL} \\ \mathbf{H}_{LC} & \mathbf{H}_{LL} \end{bmatrix} \tag{4.17}$$

The symmetric **C** and **L** submatrices are diagonally dominant descriptions of the capacitance and inductance portions of the circuit. The **H** matrix in (4.17) is merely the hybrid characterization of the dc circuit that results upon zeroing all original independent sources and forming ports appropriately for the energy storage elements.

Similarly, the input vector **b** takes the form [4.2]:

$$\mathbf{b} = \begin{bmatrix} \mathbf{C} & \mathbf{0} \\ \mathbf{0} & \mathbf{L} \end{bmatrix}^{-1} \mathbf{B} \begin{bmatrix} \mathbf{i}_I \\ \mathbf{v}_V \end{bmatrix} \tag{4.18}$$

where **B** is a dc characterization matrix similar to the **H** matrix independent from the energy storage element values, and, $\mathbf{i}_I$ and $\mathbf{v}_V$ are the current and voltage sources, respectively.

From (4.17) it follows that $\mathbf{A}^{-1}$ is

$$\mathbf{A}^{-1} = \begin{bmatrix} \mathbf{H}_{CC} & \mathbf{H}_{CL} \\ \mathbf{H}_{LC} & \mathbf{H}_{LL} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{C} & \mathbf{0} \\ \mathbf{0} & \mathbf{L} \end{bmatrix} \tag{4.19}$$

Referring to (4.19), only the dc hybrid matrix **H** needs actually to be inverted (LU factored) to calculate the moments. Inverting **H** for a circuit such as the one in Figure 4.1 is equivalent to performing the dc analysis in Figure 4.2. All capacitances are replaced by independent current sources and all inductances by independent voltage sources.

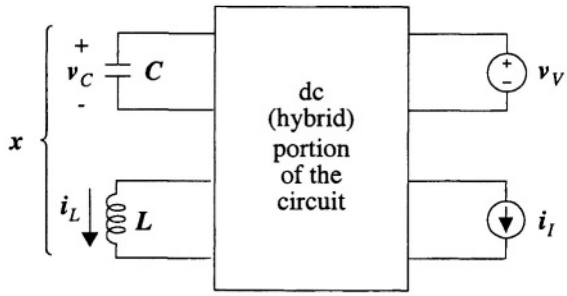


FIGURE 4.1 Circuit with energy storage elements and independent sources separated from the "dc portion".

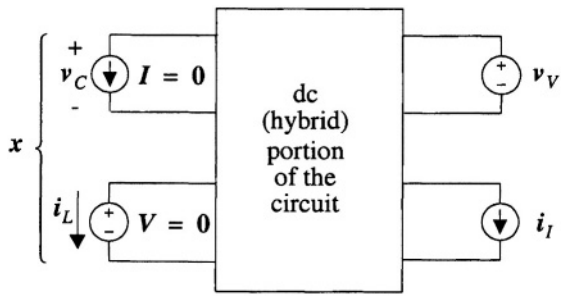


FIGURE 4.2 The dc analysis equivalent of inverting **H**.

To obtain the zeroth order moments, we solve

$$\mathbf{x}_0 = -\mathbf{A}^{-1}\mathbf{b} = -\mathbf{H}^{-1}\mathbf{B}\begin{bmatrix}\mathbf{i}_I \\ \mathbf{v}_V\end{bmatrix} \tag{4.20}$$

Equation (4.20) is equivalent to setting the independent sources, $\mathbf{v}_V$ and $\mathbf{i}_I$, equal to 1 in Figure 4.2 and solving for the open circuit capacitor voltages and short circuit inductor currents.

For higher order moments, we solve

$$\mathbf{x}_{j+1} = \mathbf{A}^{-1}\mathbf{x}_j = \mathbf{H}^{-1}\begin{bmatrix}\mathbf{C} & \mathbf{0}\\ \mathbf{0} & \mathbf{L}\end{bmatrix}\mathbf{x}_j \tag{4.21}$$

Therefore, from the same dc circuit all subsequent sets of moments can be obtained as follows:

1. Set independent sources equal to zero.
2. Set each capacitor current source equal to the product of the capacitance value C and the corresponding element of the previous moment vector $\mathbf{x}_j^C$.
3. Set each inductor voltage source equal to the product of the inductance value L and the corresponding element of the previous moment vector $\mathbf{x}_j^L$.
4. Solve for the voltages across the capacitor current sources $(\mathbf{x}_{j+1}^C)$ and the currents through the inductor voltage sources $(\mathbf{x}_{j+1}^L)$, the next set of moments.

These steps are summarized in Figure 4.3.

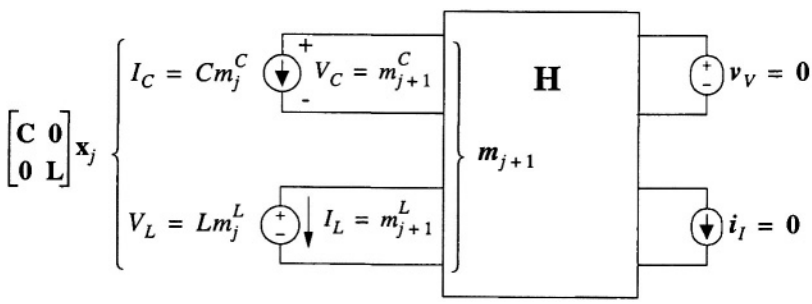


FIGURE 4.3 The dc circuit used to solve recursively for the sets of moments.

Note the similarity between the above recursion and the one described for an RC tree in Chapter 2. We need not formulate the state equations to solve the dc circuit defined in Figure 4.2 and Figure 4.3. We can use any dc circuit analysis scheme to solve for the voltages of the capacitor current sources and the currents of the inductor voltage sources. Efficient moment generation techniques for tree type interconnects and for general linear circuits are addressed in Chapter 5. We next focus on the process of moment matching.

4.3 Moment Matching

Consider the circuit function given in (4.6) again

$$H(s) = \frac{b_0 + b_1 s + \ldots + b_{n-1} s^{n-1}}{1 + a_1 s + \ldots + a_n s^n} \tag{4.22}$$

and assume that a sufficient number of its moments are known:

$$H(s) = m_0 + m_1 s + m_2 s^2 + \ldots \tag{4.23}$$

Let $\hat{H}(s)$ be a reduced order approximation for $H(s)$ in the form

$$\hat{H}(s) = \frac{\hat{b}_0 + \hat{b}_1 s + \ldots + \hat{b}_{q-1} s^{q-1}}{1 + \hat{a}_1 s + \ldots + \hat{a}_q s^q} \tag{4.24}$$

Ideally, the order of the reduced model is usually much smaller than the original one, that is,

$$q \ll n \tag{4.25}$$

Consider the moment expansion of the reduced order model:

$$\hat{H}(s) = \hat{m}_0 + \hat{m}_1 s + \hat{m}_2 s^2 + \ldots \tag{4.26}$$

The AWE technique finds the coefficients of the qth order reduced order model by matching its first $2q$ moments to the first $2q$ moments of the original circuit function,

$$\hat{m}_i = m_i \qquad 0 \le i < 2q \tag{4.27}$$

To describe how the coefficients of the reduced order model are found from the moments of the original circuit, we first show the relation between the coefficients of the reduced model with its moments.

From (4.24) and (4.26), we have

$$\hat{H}(s) = \frac{\hat{b}_0 + \hat{b}_1 s + \hat{b}_2 s^2 + \ldots + \hat{b}_{q-1} s^{q-1}}{1 + \hat{a}_1 s + \hat{a}_2 s^2 + \ldots + \hat{a}_q s^q} = \hat{m}_0 + \hat{m}_1 s + \hat{m}_2 s^2 + \ldots \tag{4.28}$$

Multiplying the denominator polynomial with the series expansion on the right hand side and then equating the terms with the same powers of s, we obtain two sets of linear equations. The first set is obtained by matching the coefficients of s^0 through s^{q-1}:

$$\begin{aligned} \hat{b}_0 &= m_0 \\ \hat{b}_1 &= m_1 + m_0 \hat{a}_1 \\ \hat{b}_2 &= m_2 + m_1 \hat{a}_1 + m_0 \hat{a}_2 \\ &\vdots \\ \hat{b}_{q-1} &= m_{q-1} + m_{q-2} \hat{a}_1 + \ldots + m_0 \hat{a}_{q-1} \end{aligned} \tag{4.29}$$

The second set is obtained by matching the coefficients of s^q through s^{2q-1}:

$$\begin{aligned} 0 &= m_q + m_{q-1} \hat{a}_1 + \ldots + m_0 \hat{a}_q \\ 0 &= m_{q+1} + m_q \hat{a}_1 + \ldots + m_1 \hat{a}_q \\ &\vdots \\ 0 &= m_{2q-1} + m_{2q-2} \hat{a}_1 + \ldots + m_{q-1} \hat{a}_q \end{aligned} \tag{4.30}$$

Note that in the above equation systems we also have used (4.27), that is, the circuit function's moments are replaced with the actual circuit moments. The only unknowns in the second set of equations are, therefore, the q coefficients of the denominator polynomial. This set can be rewritten in the matrix form as

$$\begin{bmatrix} m_{q-1} & m_{q-2} & \cdots & m_0 \\ m_q & m_{q-1} & \cdots & m_1 \\ \vdots & \vdots & \ddots & \vdots \\ m_{2q-2} & m_{2q-3} & \cdots & m_{q-1} \end{bmatrix} \begin{bmatrix} \hat{a}_1 \\ \hat{a}_2 \\ \vdots \\ \hat{a}_q \end{bmatrix} = - \begin{bmatrix} m_q \\ m_{q+1} \\ \vdots \\ m_{2q-1} \end{bmatrix} \tag{4.31}$$

The matrix on the left hand side is referred to as the *moment matrix.* Upon solving the coefficients of the denominator polynomial from (4.31), the coefficients of the numerator polynomial can be calculated by simple substitution in (4.29).

4.3.1 Poles and Residues

Once the reduced-order circuit function in the form of (4.24) is obtained, the poles, $\hat{p}_i$, of the reduced-order model can be determined by finding the roots of the denominator polynomial. Similarly, the zeros are the roots of the numerator polynomial. Although the pole-zero representation is very useful in the analysis and design of analog circuits, for time-domain interconnect analysis the pole-residue representation is preferable. After the poles are obtained, the residues, $\hat{k}_i$, are calculated using partial fraction decomposition, which allows the reduced-order model to be expressed as

$$\hat{H}(s) = \sum_{i=1}^{q} \frac{\hat{k}_i}{s - \hat{p}_i} \tag{4.32}$$

The residues can also be calculated directly from the moments. As shown in the following sections, this capability is important for some applications, especially for stability enforcement. Consider the moment expansion of the reduced order function with pole-residue representation:

$$\sum_{i=1}^{q} \frac{\hat{k}_i}{s - \hat{p}_i} = -\sum_{i=1}^{q} \frac{\hat{k}_i}{\hat{p}_i} - s\sum_{i=1}^{q} \frac{\hat{k}_i}{\hat{p}_i^2} - s^2\sum_{i=1}^{q} \frac{\hat{k}_i}{\hat{p}_i^3} - \ldots \tag{4.33}$$

or

$$\hat{m}_j = -\sum_{i=1}^{q} \frac{\hat{k}_i}{\hat{p}_i^{j+1}} \tag{4.34}$$

Matching the first q moments of the reduced order model to those of the original circuit, we obtain a linear equation system for the residues:

$$\begin{bmatrix} \hat{p}_1^{-1} & \hat{p}_2^{-1} & \cdots & \hat{p}_q^{-1} \\ \hat{p}_1^{-2} & \hat{p}_2^{-2} & \cdots & \hat{p}_q^{-2} \\ \vdots & \vdots & \ddots & \vdots \\ \hat{p}_1^{-q} & \hat{p}_2^{-q} & \cdots & \hat{p}_q^{-q} \end{bmatrix} \begin{bmatrix} \hat{k}_1 \\ \hat{k}_2 \\ \vdots \\ \hat{k}_q \end{bmatrix} = - \begin{bmatrix} m_0 \\ m_1 \\ \vdots \\ m_{q-1} \end{bmatrix} \tag{4.35}$$

With the approximate impulse response described by the closed form expression given in (4.32), the response to any kind of input can be determined easily as explained in Chapter 2. Similarly, the ac response is computed by simply evaluating (4.32) or (4.24) at the desired frequency points.

4.4 Practical Considerations

We will first present examples to demonstrate the time- and frequency-domain convergence properties of AWE, then use these results to discuss some practical issues related to the application of AWE for interconnect circuit problems.

The first example is a small RC circuit shown in Figure 4.4. It has only four poles, which can be found by writing its state equation as in (4.1) and solving for the eigenvalues of **A**. Table 2.1 shows the exact poles and the poles of the AWE approximations from the first order to the forth order. It can be observed that the approximate poles converge to the dominant poles very quickly. When the number of AWE poles is equal to the system order, approximate and exact poles are exactly the same. This is

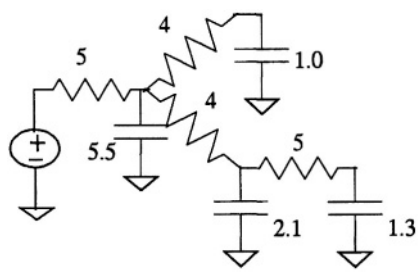


FIGURE 4.4 A small RC circuit example.

due to the convergence property of the Padé approximation which is explained in Subsection 4.5.3.

TABLE 2.1. Pole convergence for the circuit in Figure 4.4.

Exact poles	1st order	2nd order	3rd order	4th order
-0.017630	-0.014368	-0.017636	-0.017630	-0.017630
-0.095216		-0.091104	-0.095331	-0.095216
-0.287150			-0.514670	-0.287150
-0.345410				-0.345410

The second example is a large RC clock tree with 765 nodes. The response is measured at a node close to the source, since that is usually where the behavior of the signal is "stiff." The stiff responses are known to be difficult to approximate with reduced order models. The time-domain waveforms of the first, second, and third order AWE approximations along with the SPICE simulation result are shown in Figure 4.5. The second and the third order approximations are indistinguishable from the SPICE waveform.

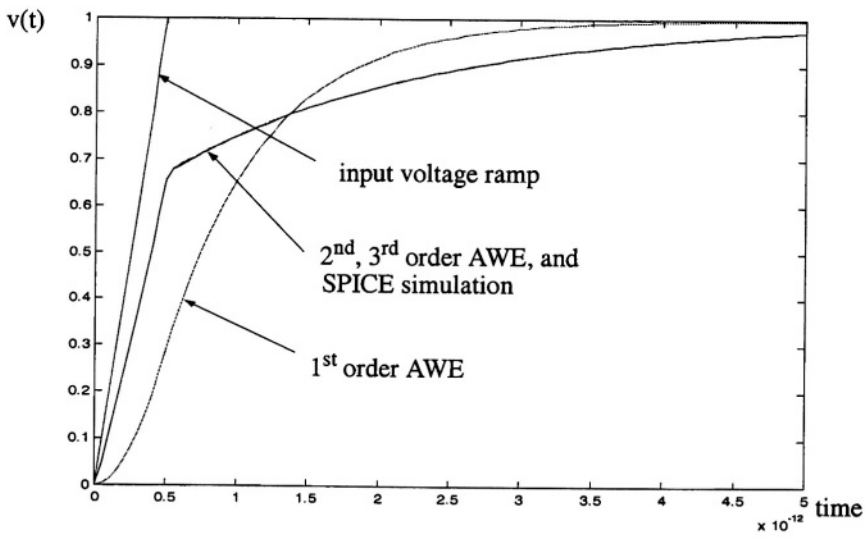


FIGURE 4.5 Time-domain response of an RC tree.

Another example is an RLC clock tree from an industrial IC. It has 1497 nodes, 344 inductors, and 658 capacitors. Different order AWE approximations at the output node are shown in Figure 4.6 together with the SPICE simulation result. Notice that the first order approximation is very inaccurate since the first two moments have no inductance information. The third order approximation and beyond -- up to exact order or the maximum numerical resolution -- match the SPICE simulation exactly.

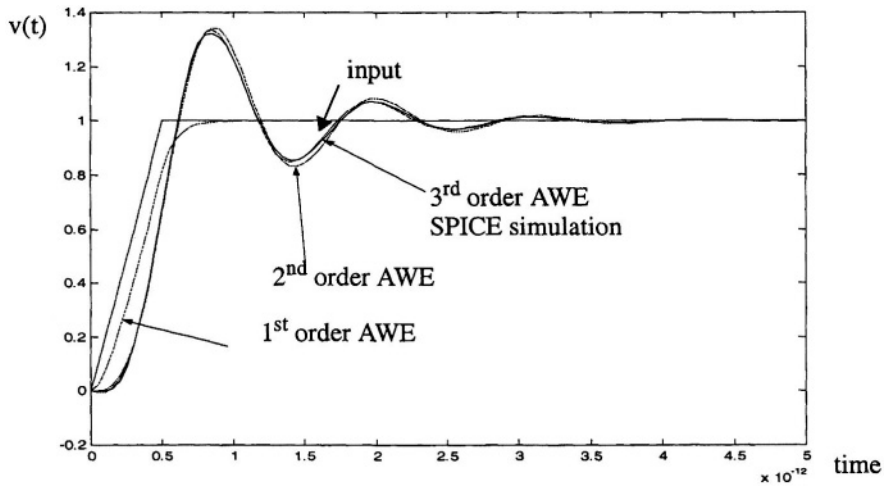


FIGURE 4.6 Time-domain response of an RLC clock tree.

As evident from the above examples, in most cases AWE works very well and provides very good approximations both in the time and frequency domains. The reason why AWE provides good waveform approximations can be explained as follows: AWE matches the moments of the original circuit to a reduced order model. Since the moments are the coefficients of the Taylor series expansion around $s = 0$, the reduced order model is expected to be a good approximation at low frequencies, and exact at $s = 0$. Importantly, most interconnect structures, like all physical systems, exhibit a low-pass filter behavior -- they attenuate the high frequency components of the signal as it propagates along the system. Digital input signals are generally modeled as combinations of step and ramp functions, implying that they have a rapidly decaying frequency spectra and a pulse propagation that is totally dominated by the low-frequency behavior. Hence, AWE produces excellent waveform analysis results and sometimes even better results than a SPICE simulation which incurs some small amount of numerical integration error.

AWE can sometimes fail, however, either in terms of unstable poles or simply by way of inaccurate results. As we will see in the next section, AWE finds Padé approximations for circuit-response functions. It is partially the Padé approximation and partially the explicit moment matching on a finite precision computer that are responsible for the limitations of AWE. We will address these limitations in more detail later. But for now, consider that the accuracy limitation occurs simply because of the precision loss during numerical computations.

The stability of AWE can be a substantial problem. In the presence of positive poles, the time-domain response of the approximation becomes unbounded, and therefore such an approximation is useless. There are two types of instability:

1. *Padé approximation instability:* This type of instability is inherent to the Padé approximation. It is independent of the numerical errors introduced during the computations.
2. *Moment-matching instability:* This type of instability is due to precision loss that occurs during the moment calculations.

We next establish the theoretical connection between AWE and Padé approximations. We then describe moment-matching related limitations of AWE and propose methods to overcome both types of instability problems.

4.5 Padé Approximation

The approximation method used in AWE is actually a specific type of a well-known rational function approximation -- *Padé approximation* [4.3]. Padé approximation has long been a subject of interest in numerical analysis, and has been widely used in many different engineering and science fields. It has been especially popular in the area of model order reduction as it applies to control theory.

The definition of Padé approximation is as follows. Assume a rational function

$$R_{n,m}(x) = \frac{B_n(x)}{A_m(x)} \tag{4.36}$$

where $B_n(x)$ and $A_m(x)$ are polynomials of order n and m, respectively. Then $R_{n,m}(x)$ is a Padé approximation of type [n/m] to a function $F(x)$, if the first $n + m + 1$ terms of its Taylor expansion equal to the Taylor expansion of $F(x)$.

4.5.1 Calculation of Padé Approximation Coefficients

We now show how the coefficients of Padé approximation are calculated from the Taylor series coefficients (moments) of the original function. In the previous section, this is derived for $[(q-1)/q]$ type AWE approximations. Now we generalize it to $[n/m]$ type approximations.

Assume that the Taylor series expansion of the function $F(x)$ is known:

$$F(x) = m_0 + m_1 x + m_2 x^2 + \dots , \tag{4.37}$$

The goal is to find a Padé approximation of type $[n/m]$ of the function $F(x)$ in the form

$$R_{n,m}(x) = \frac{b_0 + b_1 x + b_2 x^2 + \dots + b_n x^n}{1 + a_1 x + a_2 x^2 + \dots + a_m x^m} \tag{4.38}$$

whose Taylor series agrees with the Taylor series of $F(x)$ in the first $n + m + 1$ terms. To find the coefficients, we write

$$\frac{b_0 + b_1 x + b_2 x^2 + \dots + b_n x^n}{1 + a_1 x + a_2 x^2 + \dots + a_m x^m} = m_0 + m_1 x + \dots + m_{m+n} x^{m+n} + \dots \tag{4.39}$$

By multiplying the denominator polynomial with the right hand side and matching the coefficients of s^0 through s^{m+n}, we obtain two sets of equations for the coefficients.

The first one gives the coefficients of the denominator polynomial,

$$\begin{bmatrix} m_n & m_{n-1} & \cdots & m_{n-m+1} \\ m_{n+1} & m_n & \cdots & m_{n-m+2} \\ \vdots & \vdots & \ddots & \vdots \\ m_{n+m-1} & m_{n+m-2} & \cdots & m_n \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_m \end{bmatrix} = - \begin{bmatrix} m_{n+1} \\ m_{n+2} \\ \vdots \\ m_{n+m} \end{bmatrix} \tag{4.40}$$

where $m_k = 0$ for $k < 0$. Then the numerator coefficients are obtained from

$$\begin{aligned} b_0 &= m_0 \\ b_1 &= m_1 + m_0 a_1 \\ &\vdots \\ b_n &= m_n + m_{n-1} a_1 + \ldots + m_0 a_n \end{aligned} \tag{4.41}$$

where $a_k = 0$ for $k > m$.

4.5.2 Padé Approximation Instability

An important drawback associated with AWE is the inherent Padé approximation instability. It is well known that Padé approximations of stable systems may be unstable [4.4]. To demonstrate this, we will use a two-pole one-zero system function

$$H(s) = \frac{s - z}{(s - p_1)(s - p_2)} \tag{4.42}$$

where the poles p_1 and p_2 are negative real numbers. Let us consider a first order Padé approximation of type [0/1] to $H(s)$,

$$\hat{H}(s) = \frac{\hat{k}}{s - \hat{p}} \tag{4.43}$$

After some manipulation it can be shown that

$$\hat{p} = \left(\frac{1}{p_1} + \frac{1}{p_2} - \frac{1}{z} \right)^{-1} \tag{4.44}$$

which reveals an interesting property of Padé approximation: the approximate poles depend not only on the actual poles, but also on the actual zeros. Even though both of the actual poles are negative numbers, it is easy to find values of p_1, p_2, and z, which would make $\hat{p}$ positive, implying an unstable approximation to a stable system.

In higher order systems, such anomalies are not limited to the first order approximations, but can occur at any order. There exists a wealth of literature investigating this problem in the control theory area, and there have been a variety of modifications of Padé approximations aimed at avoiding unstable poles. For some references, see [4.4]. Most of these methods, however, are not applicable to interconnect analysis problems for reasons that are beyond the scope of this book.

Although instability can occur at any order, fortunately in most cases it disappears by increasing the order of the approximation. To demonstrate this consider a small RC circuit. The poles and the corresponding residues from various orders of approximations are shown in Table 2.2 . It can be observed that the second order approximation has one unstable pole with a non-negligible residue. This indicates that this instability is not due to a numerical error but is a product of the Padé approximation. The unstable pole disappears in the third and forth order approximations,

TABLE 2.2. The poles and residues of the various order AWE approximations for a small RC circuit.

1st order		2nd order		3rd order		4th order	
poles	residues	poles	residues	poles	residues	poles	residues
-1.7346e-3	1.7346e-3	**9.2563e-2**	**-2.4240e-3**	-3.0084e-2	1.5471e-3	-1.5066e-1	-5.7977e-3
		-1.6884e-3	1.6441e-3	-1.0276e-2	-2.7464e-4	-5.8923e-2	4.8272e-3
				-1.6891e-3	1.6474e-3	-9.2375e-3	-1.7320e-4
						-1.6892e-3	1.6474e-3

4.5.3 Convergence

From the definition of Padé approximation, it follows that a qth order standard AWE model is a Padé approximation of type $[(q-1)/q]$ to the original circuit function. Such an approximation, therefore, has $q-1$ zeros and q poles. However, the flexibility of having any number of poles and zeros provides some advantages to overcome some of the limitations of AWE. The entire array of approximations that are possible with different choices of n and m is known as the Padé table:

$$\begin{matrix} R_{0,0} & R_{0,1} & \cdots & R_{0,m} & \cdots \\ R_{1,0} & R_{1,1} & \cdots & R_{1,m} & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ R_{n,0} & R_{n,1} & \cdots & R_{n,m} & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \end{matrix} \tag{4.45}$$

The entire Padé table can be searched to find the best Padé approximation in terms of accuracy, provided that it is a stable one, but this can be time consuming [4.4] [4.5]. Two practical popular techniques to search the Padé table are the diagonal sequence

and the horizontal sequence. The horizontal sequence is usually applied to improve stability. This technique is also known as moment shifting and addressed in Section 4.6.3.

The diagonal sequence searches the Padé table along a subdiagonal, typically the subdiagonal which has one pole more than zeros ($[q-1/q]$ type approximations), due to its resemblance to actual circuit functions. The original form of AWE approximation is of this type. This type of approximation generally approaches the actual circuit function as the order of approximation increases. This fact is stated in the following theorem [4.6]. The theorem is given for the main diagonal sequence, but it is equally applicable to the subdiagonal sequences.

Theorem 4.1: *Let $H(s)$ be a meromorphic function. Let ε and δ be two given positive numbers. Then q_0 exists such that any $R_{q,q}(s)$ Padé approximation satisfies*

$$\left|H(s) - R_{q,q}(s)\right| < \varepsilon \tag{4.46}$$

for all $q > q_0$ in any domain $|s| \le R$, $R > 0$ except for a set of areas less than δ.

This theorem has several implications. First it proves that the diagonal Padé sequence converges uniformly to the actual circuit function in a disc, except for some areas, or sets of complex numbers. Furthermore, the theorem states that the total area of such exceptional sets can be made arbitrarily small by increasing the order.

The exceptions in the above theorem correspond to either the poles of the original function or the poles of the approximation. Since $H(s)$ becomes unbounded in the vicinity of an actual pole, the inequality in the theorem cannot hold there. The fact that the area of an exception set can be made decreasingly small implies that sufficiently high order approximations contain a pole approaching the actual pole. In addition, due to uniform convergence, all of the approximations above a certain order must contain that pole. This proves that the approximate poles corresponding to actual poles appear persistently.

Another type of exception set centers around those approximate poles that do not correspond to any actual poles; namely, defective poles. Since their areas can be made exceedingly small, the theorem implies that the residues of the corresponding approximate poles should vanish as the order of approximation increases.

4.6 Moment-Matching Issues

In addition to the inherent Padé problems, AWE has limitations due to the numerical computation noise that occurs during moment calculation and moment matching. AWE suffers from these numerical noise problems because it employs an *explicit* moment matching to find Padé approximation. Later in this book we will see other methods to compute Padé approximations of the circuit responses. These methods reduce the numerical noise related problems by directly computing the Padé approximation instead of applying moment matching.

4.6.1 Solving the Moment Matrix

A critical step in AWE is inverting the moment matrix in (4.31). The existence and quality of a Padé approximation depends on the invertibility and condition of the corresponding moment matrix. Its condition number is a measure of the round-off errors introduced during moment calculation. An ill-conditioned matrix can still be inverted and it does not always mean that the approximation will be totally bogus, but it most probably indicates that we are trying to extract more poles than those that are numerically prominent in the moments.

A. Scaling

The condition of the moment matrix can be improved to some extent by scaling the moments. Consider a moment expansion

$$H(s) = m_0 + m_1 s + m_2 s^2 + \ldots \tag{4.47}$$

The rate in which the moments change can be speculated from (4.34). If λ approximately represents the order of the magnitude of the dominant poles, we can claim that

$$|m_i / m_{i+1}| \approx \lambda \tag{4.48}$$

The above equation indicates that the range of the numbers in a moment matrix can be very big depending on the order of the dominant poles, thus making the matrix ill-conditioned.

Introducing a new frequency variable s', such that $s' = s/\lambda$, the moments in (4.47) become

$$H(s') = m_0 + \lambda m_1 s' + \lambda^2 m_2 s'^2 + \ldots \tag{4.49}$$

Obviously, frequency scaling keeps the rate of the change in the moments reasonable. Such a scaling is achieved by multiplying all capacitance and inductance values by λ. After the approximation is obtained, we need to scale the frequency back by multiplying poles and residues by λ.

Although the exact locations of the poles are not known before the moment calculation, their order can be estimated from the circuit. Typically $\lambda = 10^9$ is a good scaling value so that 1 GHz becomes 1 Hz and 1 ns becomes 1 second.

The amount of scaling can also be chosen as the ratio of the first two moments:

$$\lambda = \left| \frac{m_0}{m_1} \right| \tag{4.50}$$

However, we must admit that scaling provides only minimal help and other steps must be taken to solve the problem completely.

4.6.2 Dominant Pole Convergence

One of the important peculiarities of Padé approximation is the dominant pole convergence property. It effects both stability and accuracy of the approximation, but can also be used to improve the stability via moment shifting.

To explain the dominant pole convergence property, we start with the relation between the moments and the poles and residues of the original circuit:

$$m_j = -\sum_{i=1}^{n} \frac{k_i}{p_i^{j+1}} \tag{4.51}$$

Assume that the poles are ordered with increasing magnitudes: $|p_1| \le |p_2| \le \ldots \le |p_n|$. In order to see how the moments are effected from the individual poles, we rewrite (4.51) in the form

$$m_j = \frac{k_1 + k_2\left(\frac{p_1}{p_2}\right)^{j+1} + \ldots + k_2\left(\frac{p_1}{p_n}\right)^{j+1}}{p_1^{j+1}} \tag{4.52}$$

Now consider the contribution from one of the large poles, e.g., p_n. For the sake of simplicity, we assume $p_1 \neq p_n$, which implies $|p_1/p_n| < 1$. Therefore, as the order increases the portion coming from the pole p_n decreases rapidly as compared to the contribution from the smaller magnitude poles. This implies that higher order moments contain little information about the larger poles, and the larger the pole the less the contribution.

The moment convergence phenomenon can also be explained in terms of circuit matrix-vector operations. Considering the state space formulation again, in terms of the state matrix **A**, the moments are given by

$$m_j = -\mathbf{c}^T\mathbf{A}^{-j-1}\mathbf{b} \tag{4.53}$$

It is a well known fact from linear algebra that as j increases, the product $\mathbf{A}^{-j-1}\mathbf{b}$ will converge to the eigenvector of the largest eigenvalue of the matrix $\mathbf{A}^{-1}$ [4.7]. Note that the largest eigenvalue of $\mathbf{A}^{-1}$ corresponds to the smallest pole of the circuit response.

So how does this moment convergence property impact the quality of an AWE approximation? First, it limits the accuracy of the approximation. Ideally one expects more and more accurate approximations as the order is increased. However, since the higher order moments do not contain additional useful information, including more moments does not increase the quality of the approximation beyond some order. After that order, which is typically somewhere from 6 to 12, AWE starts generating bogus poles which have no effect on the accuracy. Furthermore, very often these bogus poles appear with positive real parts resulting in unstable approximations.

For example, consider a simple 1-bit bus line with 15 RC sections. The frequency domain plots of different approximation orders are shown in Figure 4.7. After the 6th order, there is very little change in the Bode plot which means there will not be a change in the time-domain response by using more poles. All of the approximations up to order five in this example are stable, however the 6th, 8th, and 9th order approximations contain bogus unstable poles.

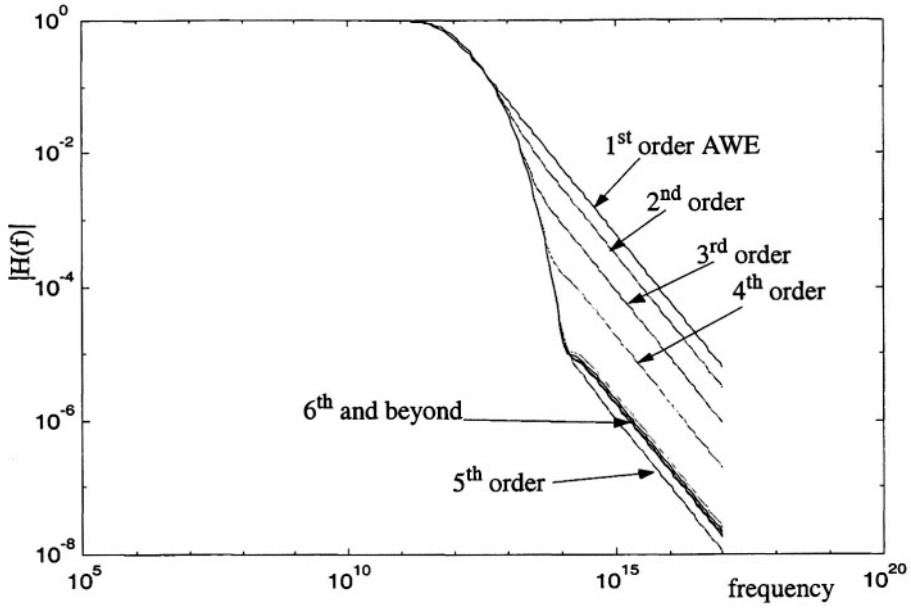


FIGURE 4.7 Bode plot of different approximation orders for a 1-bit bus example.

4.6.3 Moment Shifting

But the dominant pole convergence property can always be used to provide stability for low-order approximations. The basic idea is to use higher order moments to obtain the poles of the reduced order model. As discussed in the previous subsection, the information contained in the higher order moments is limited to the dominant poles of the actual circuit. Therefore, when the approximated poles are obtained from higher order moments they are more likely to converge to the actual dominant poles. This is an important property of Padé approximation and a more formal description is as follows:

Theorem 4.2: *Let H(s) be regular in the domain* $|s| < R$, *except for a finite number of poles inside the domain, of total multiplicity N. Then the Padé approximation sequence* $\hat{H}_{i,N}(s)$, *with N fixed, converges uniformly to H(s) for* $|s| < R$, *except at the poles of* $\hat{H}_{i,N}(s)$ *and H(s).*

Therefore, with a horizontal sequence, since the approximate poles converge to the actual poles, the approximation is less likely to generate unstable poles. The details of this procedure are as follows.

In Section 4.4, we defined Padé approximations with arbitrary numerator and denominator orders. To find the poles of an [*n/m*] type approximation we need 2*m* moments to be used in the linear equation system given in (4.40) and we use the set $\{m_{n-m+1}, m_{n-m+2}, \ldots, m_{n+m}\}$. Since the choice of *n* is arbitrary, any set of 2*m* successive moments can be used to find *m* approximated poles. In conventional AWE approximation, for instance, the first 2*m* moments, $\{m_0, m_1, \ldots, m_{2m-1}\}$, are used.

This procedure is referred to as *moment shifting* in [4.8], where it is shown to be equivalent to exciting the circuit with signals with less high frequency components, and thereby eliminating the effects of the high-frequency poles which can cause instability. One shift, for example, corresponds to step input excitation. One shift often yields a stable approximation, but if does not, shifting is allowed to continue until a stable solution is obtained or until shifting is no longer possible due to numerical limitations. Once the stable poles are found via shifting, their residues are obtained using (4.35).

We should note that moment shifting does not necessarily guarantee stability, but it has proven to be extremely reliable and useful in practice. To illustrate moment shifting, consider a specific case: a single pole approximation. In conventional AWE, the approximation takes the form

$$\hat{H}(s) = \frac{\hat{b}_0}{1 + \hat{a}_1 s} = \frac{\hat{k}_1}{s - \hat{p}_1} \tag{4.54}$$

The coefficient $\hat{a}_1$ is given by

$$\hat{a}_1 = \frac{-m_1}{m_0} \tag{4.55}$$

which implies

$$\hat{p}_1 = \frac{m_0}{m_1} \tag{4.56}$$

In some RLC circuit responses it is possible for m_0 and m_1 to have the same signs, and therefore unstable one pole approximations can occur. However, moment shifting allows us to search for a stable first order approximation. For a general [n/1] Padé approximation, the coefficient $\hat{a}_1$ is given by $\hat{a}_1 = -m_n/m_{n-1}$. Thus

$$\hat{p}_1 = \frac{m_{n-1}}{m_n} \tag{4.57}$$

So, n is increased until a stable pole is obtained. On the other hand, using (4.52) it can be easily shown that the ratio of two successive moments converges to the most dominant pole:

$$\lim_{j \to \infty} \frac{m_j}{m_{j+1}} = p_1 \tag{4.58}$$

where it is assumed that p_1 is real. Thus the approximated pole given in (4.57) converges to the actual dominant pole of the circuit as n increases.

For the bus example used in the previous subsection, the ratio in (4.57) for increasing values of n is listed in Table 2.3 . The ratio rapidly converges to the most dominant pole of that circuit, which is at -4.5879e+12.

TABLE 2.3. Moment ratio convergence to the most dominant pole.

j	m_j	m_{j+1}	m_j/m_{j+1}
0	1.0000e+000	-2.5538e-013	-3.9158e+012
1	-2.5538e-013	5.6633e-026	-4.5093e+012
2	5.6633e-026	-1.2366e-038	-4.5796e+012
3	-1.2366e-038	2.6959e-051	-4.5871e+012
4	2.6959e-051	-5.8762e-064	**-4.5879e+012**
5	-5.8762e-064	1.2808e-076	**-4.5879e+012**
6	1.2808e-076	-2.7916e-089	**-4.5879e+012**

The rate of convergence to the dominant poles is established in the following theorem [4.9]:

Theorem 4.3: *Define* σ *and* R *such that* $0 < |p_1| \le |p_2| \le \ldots \le |p_N| < \sigma R < R$, *where* p_i*'s are the actual poles of a circuit response such that* $|p_N| \ne |p_{N+1}|$. *Let*

$$F(s) = \left(1 - \frac{s}{p_1}\right)\left(1 - \frac{s}{p_2}\right)\ldots\left(1 - \frac{s}{p_N}\right)$$
$$= 1 + f_1 s + f_2 s^2 + \ldots + f_N s^N$$

Let $a_j^{(i)}$*'s be the coefficients of* $A_{i,N}(s)$, *the denominator polynomial of* $\hat{H}_{i,N}(s)$:

$$A_{i,N}(s) = 1 + a_1^{(i)} s + a_2^{(i)} s^2 + \ldots + a_N^{(i)} s^N$$

Then

$$a_j^{(i)} = f_j + O(\sigma^i) \qquad \text{and} \qquad A_{i,N}(s) = F(s) + O(\sigma^i).$$

The quantity σ is a measure of the relative difference between $|p_N|$ and $|p_{N+1}|$:

$$\frac{|p_N|}{|p_{N+1}|} \le \sigma < 1 \tag{4.59}$$

According to the above theorem, the rate of convergence depends on σ. Therefore, the larger the ratio $|p_{N+1}|/|p_N|$, the faster the convergence to the actual poles.

4.6.4 Frequency Shifting

The dominant pole convergence property can be used to obtain stable low-order approximations. The same property, on the other hand, limits the accuracy that can be obtained from an approximation. As we have seen previously, the higher order moments are dominated by the smaller poles, and consequently, beyond some order, the quality of the approximation does not improve. As Theorem 4.3 implies, this limitation is more pronounced if there is a big spread in the ratio of the poles. Frequency shifting alleviates this problem to some degree.

To understand the shifting, refer to Figure 4.8. Initially the ratio p_1/p_2 is a very large number which may cause numerical problems in AWE. Our objective is to reduce the

ratio. This can be achieved by shifting all of the poles to the left or moving the *jw* axis to the right, which is illustrated in Figure 4.8. As an example, consider a circuit which has poles $p_1 = -1$ and $p_2 = -1000$. If they are shifted by 1000 to the left, p_1 and p_2 become -1001 and -2000, respectively, and their ratio reduces from 1000 to approximately 2. Now the dominance of the smallest pole is decreased which in turn causes the moments to contain more information about the initially non-dominant poles.

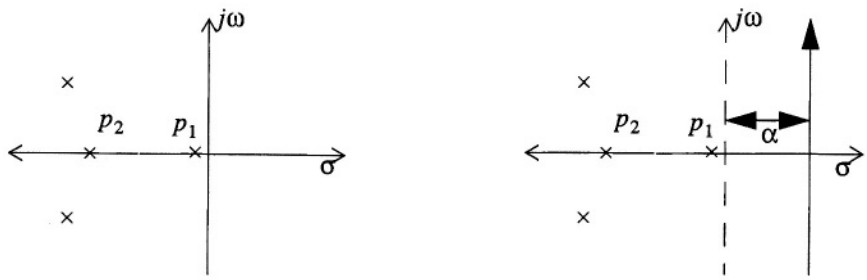


FIGURE 4.8 Frequency shifting.

In circuits, shifting is achieved by adding a parallel conductance to each capacitance and a series resistance to each inductance as shown in Figure 4.9. Let us introduce a new Laplace variable s', such that

$$s' = s - \alpha \tag{4.60}$$

where α is the shift amount. In the shifted domain, we have

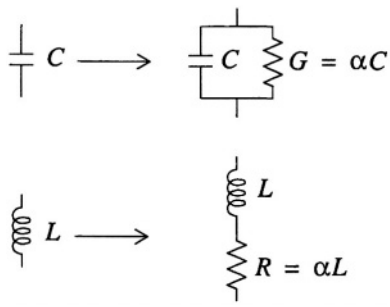


FIGURE 4.9 Capacitor and inductor under frequency shifting.

$$sC \rightarrow s'C + \alpha C \qquad sL \rightarrow s'L + \alpha L \tag{4.61}$$

The moments around $s' = 0$ $(s = \alpha)$ can be calculated in a similar manner to standard AWE. Frequency shifting usually improves the quality of the approximation, but it comes with additional computational cost. Because of the parallel conductances added to capacitances, interconnect circuits can no longer be modeled as trees, and therefore the methods such as path tracing cannot be applied efficiently. Perhaps, the best way to compute frequency shifted moments is using modified nodal analysis formulation as will be explained in the next chapter.

By applying the standard moment matching to shifted moments, we obtain the poles and their residues in the form

$$H(s') = \sum_i \frac{\hat{k}_i'}{s' - \hat{p}_i'} \tag{4.62}$$

or shifting back

$$H(s) = \sum_i \frac{\hat{k}_i}{s - \hat{p}_i} \tag{4.63}$$

where $\hat{p}_i = \hat{p}_i' + \alpha$ and $\hat{k}_i = \hat{k}_i'$.

An important practical issue with the frequency shifting is the amount of the shift or the expansion frequency α. It must be obvious that it is impossible to find an optimal shift amount, since this would require the original pole distribution on the complex plane. Practically, the expansion point should be chosen such that from that point the spread in the distances to the poles in the desired bandwidth should be minimized. From this perspective, the shift amount should be comparable with the bandwidth. For example, α can be selected as:

$$\alpha = 2\pi f_{\max} \tag{4.64}$$

where $f_{\max}$ is the maximum frequency of interest. A shift much smaller than this may not provide the effect that we seek. Importantly, an expansion too far away from the origin can generate totally noisy poles because it may be difficult for AWE to find any poles on which to converge.

To demonstrate the effect of the frequency shifting, we consider a circuit with a dominant first pole and compute four AWE approximation poles for various values of α. The results are plotted in Figure 4.10. The first pole is not affected from shifting until α reaches 1e+12. But the quality of the second AWE pole is dependent on the α value. It is a good approximation to the second pole of the circuit when α is between 2e+9 and 2e+11, and it is almost random elsewhere.

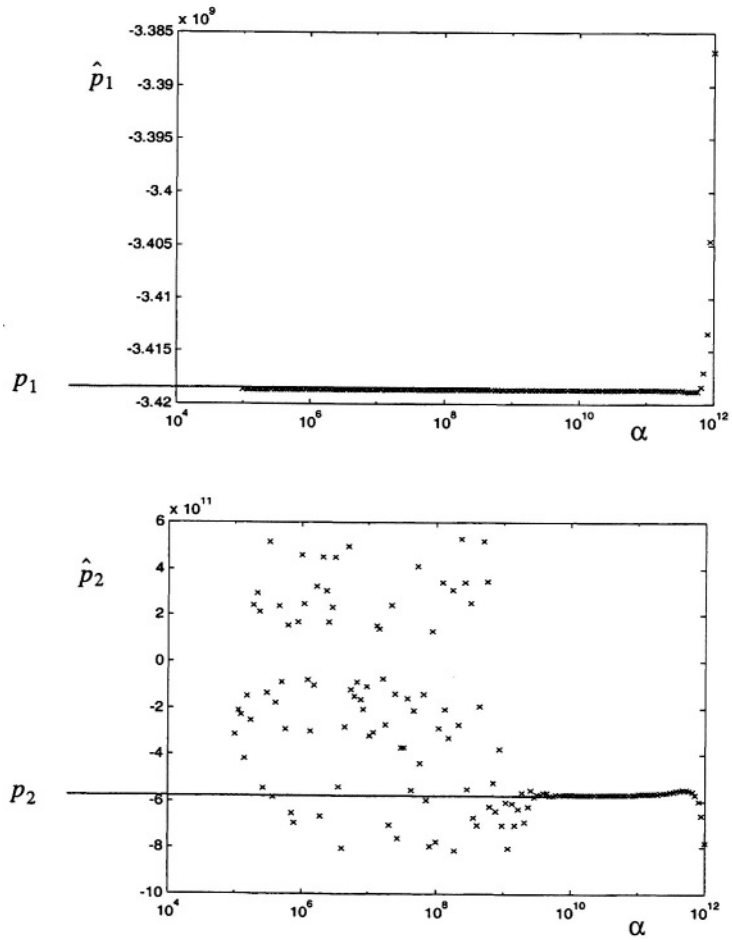


FIGURE 4.10 The first two AWE poles, $\hat{p}_1$ and $\hat{p}_2$, vs shift amount (exact poles: $p_1 = -3.4187e+9$, $p_2 = -5.7640e+11$).

4.6.5 Expansions about $s = \infty$

Another issue with AWE is that using expansions about $s = 0$ would produce large time-domain errors near the initial time point ($t = 0$). This is due to the facts that the moments obtained at $s = 0$ contain mostly low-frequency information and that initial time-domain response near $t = 0$ is mainly determined by high-frequency behavior of the circuit. The accuracy near initial time, therefore, can be improved by including circuit information about $s = \infty$ to the approximation [4.10]. For this consider the power series expansion of the circuit function around $s = \infty$:

$$H(s) = \frac{m_{-1}}{s} + \frac{m_{-2}}{s^2} + \frac{m_{-3}}{s^3} + \dots \tag{4.65}$$

The coefficients of the above expansion are called *Markov parameters* [4.11]. They are also known as *derivatives* in the literature because they are related to the time derivatives of the impulse response at $t = 0$:

$$m_{-j} = \left.\frac{d^j h(t)}{dt}\right|_{t=0} \tag{4.66}$$

The derivatives can be computed recursively similar to the moment computation. This can be seen by considering the power series expansion of the state vector $\mathbf{X}(s)$ around $s = \infty$

$$\mathbf{X}(s) = \mathbf{x}_{-1}s^{-1} + \mathbf{x}_{-2}s^{-2} + \dots \tag{4.67}$$

where $\mathbf{x}_{-j} = \mathbf{A}^{j-1}\mathbf{b}$. From (4.5) it follows that $m_{-j} = \mathbf{c}^T\mathbf{x}_{-j}$.

A. Mixed Moment and Markov Parameter Matching

Previously, to find the coefficients, $\hat{a}_i$'s and $\hat{b}_i$'s in (4.24), of the reduced model, we used the first $2q$ moments of the original circuit. Alternatively, the reduced order model can be constructed by matching the leading r moments and the leading $2q - r$ Markov parameters:

$$\frac{\hat{b}_0 + \hat{b}_1 s + \ldots + \hat{b}_{q-1} s^{q-1}}{1 + \hat{a}_1 s + \ldots + \hat{a}_q s^q} = m_0 + m_1 s + m_2 s^2 + \ldots$$
$$= m_{-1} s^{-1} + m_{-2} s^{-2} + m_{-3} s^{-3} + \ldots \tag{4.68}$$

This 2-point matching yields the following linear equation set for the $\hat{a}_i$'s:

$$\begin{bmatrix} m_{-(q-r+1)} & m_{-(q-r+2)} & \cdots & m_{-(2q-r)} \\ m_{-(q-r)} & m_{-(q-r+1)} & \cdots & m_{-(2q-r-1)} \\ \vdots & \vdots & \ddots & \vdots \\ m_{r-2} & m_{r-3} & \cdots & m_{-(q-r+1)} \end{bmatrix} \begin{bmatrix} \hat{a}_1 \\ \hat{a}_2 \\ \vdots \\ \hat{a}_q \end{bmatrix} = - \begin{bmatrix} m_{-(q-r)} \\ m_{-(q-r-1)} \\ \vdots \\ m_{r-1} \end{bmatrix} \tag{4.69}$$

The $\hat{b}_i$'s are then computed as

$$\begin{aligned} \hat{b}_0 &= m_0 \\ \hat{b}_1 &= m_1 + m_0 \hat{a}_1 \\ &\vdots \\ \hat{b}_{r-1} &= m_{r-1} + m_{r-2} \hat{a}_1 + \ldots + m_0 \hat{a}_{r-1} \end{aligned} \tag{4.70}$$

and

$$\begin{aligned} \hat{b}_{q-1} &= m_{-1} \hat{a}_q \\ \hat{b}_{q-2} &= m_{-2} \hat{a}_q + m_{-1} \hat{a}_{q-1} \\ &\vdots \\ \hat{b}_r &= m_{-(q-r)} \hat{a}_q + \ldots + m_{-1} \hat{a}_{r+1} \end{aligned} \tag{4.71}$$

Derivatives have proven to be useful in piecewise linear transient simulation since it is important to match initial conditions accurately [4.12]. In general, however, the advantages over pure moment matching are not obvious. Also, it is not clear how many derivatives to match. Perhaps the most useful technique is to include just the first derivative to match the initial conditions at $t = 0$.

4.6.6 Partial Padé Approximation

After trying all of these stabilizing methods we may still have unstable poles or we may want to proceed with the original set of poles of which some are unstable. In these cases, we simply discard all of the unstable poles. We then find the residues of the stable poles by solving the residue equation given in (4.35):

$$\begin{bmatrix} p_1^{-1} & p_2^{-1} & \cdots & p_{q'}^{-1} \\ p_1^{-2} & p_2^{-2} & \cdots & p_{q'}^{-2} \\ \vdots & \vdots & \ddots & \vdots \\ p_1^{-q'} & p_2^{-q'} & \cdots & p_{q'}^{-q'} \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \\ \vdots \\ k_{q'} \end{bmatrix} = - \begin{bmatrix} m_0 \\ m_1 \\ \vdots \\ m_{q'-1} \end{bmatrix} \tag{4.72}$$

The order of the approximation is now q', where $q' < q$.

In this case, we match the first q' moments exactly and since the poles are calculated using the first $2q$ moments, we approximately match $2q$ moments. Thus, the approximation is known as partial Padé approximation [4.13].

4.7 Multi-point AWE Methods

To overcome the accuracy limitation of the single-point moment matching, a number of multipoint AWE algorithms have been proposed. In these approaches, a merged AWE approximation is formed from multiple expansion frequencies.

A general technique to include multiple shift frequencies in an AWE approximation is described in [4.14]. The process begins by obtaining an AWE approximation with the first shift frequency. A merged approximation is formed by repeatedly deflating each subsequent set of shifted moments by the existing approximation. The deflated moment set is used to form an AWE approximation -- this set of poles and residues is added to the previous approximation as a corrector. This process continues until the energy contained in the deflated data set is sufficiently small.

In transmission line circuits, the responses cannot be completely characterized by low-frequency poles. The high-Q poles close to the imaginary axis poles also affect the signal propagation. Therefore, for these types of circuits, expansions on the imaginary axis yield more accurate approximations.

Such a technique, called complex frequency hopping (CFH) [4.15], first performs expansions about $s = 0$ and $s = j\omega_{max}$. If there exist poles that appear in both of these expansions, then the search is considered complete. Otherwise, more frequencies are selected using a binary search, and expansion is carried out until each intermediate frequency has at least one pole in common with the frequency above and below it.

Another approach is proposed in [4.16]. In this method, moments from different expansion points are matched to a single transfer function. A qth order approximation matches a total of $2q$ moments form different frequencies, hence it is a multipoint Padé approximation [4.3] to the original circuit function.

4.8 Summary

Asymptotic waveform evaluation (AWE), with its many extensions for controlling stability, provides an effective moment matching approximation methodology for interconnect circuits. Although it has clear limitations and problems, we believe that AWE is the best technique for waveform analysis for on-chip interconnection circuits --- particularly those that are modeled as RC (or RL) circuits. Even for most on-chip RLC interconnect circuits, AWE works very well. However, as the strong coupling and transmission line effects start to dominate the behavior of the waveform propagation, AWE, as a single point moment matching method, suffers from accuracy and stability problems. Circuits with strong coupling and reflection effects generally require more poles than AWE can provide.

Although we addressed the stability issues of the reduced order models we have not considered the passivity. For linear circuit analysis, passivity is not a concern. But when reduced order linear models and nonlinear devices are combined together in a circuit simulation environment, for a stable simulation, the reduced order blocks have to be passive too. In the next chapter, we present the PRIMA method which guarantees passivity and stability. Furthermore, PRIMA uses Krylov vectors instead of moments and therefore does not suffer from the numerical problems that occur in AWE.

References

[4.1] L. Pillage and R. Rohrer, "Asymptotic Waveform Evaluation for timing analysis," *IEEE Transactions on Computer-Aided Design,* vol. 9, no. 4, pp. 352-366, Apr. 1990.

[4.2] L. O. Chua and P. Lin, *Computer-Aided Analysis of Electronic Circuits: Algorithms and Computational Techniques.* Englewood Cliffs, NJ: Prentice-Hall, 1975.

[4.3] G. A. Baker, Jr. and P. Graves-Morris, *Padé Approximants, Part 1: Basic Theory.* Reading, MA: Addison Wesley, 1981.

[4.4] X. Huang, "Padé approximation of linear(ized) circuit responses," Ph.D. thesis, Carnegie Mellon University, Nov. 1990.

[4.5] E. Chiprout and M. Nakhla, *Asymptotic Waveform Evaluation and Moment Matching for Interconnect Analysis.* Boston: Kluwer, 1993.

[4.6] J. Zinn-Justin, "Strong interaction dynamics with Padé approximants," *Physics Report,* pp. 56-102, 1970.

[4.7] G. H. Golub and C. F. Van Loan, *Matrix Computations,* Third Edition. Baltimore, MD: The Johns Hopkins University Press, 1996.

[4.8] D. Anastasakis, N. Gopal, S.-Y. Kim, and L. Pillage, "Enhancing the stability of asymptotic waveform evaluation for digital interconnect circuit applications," *IEEE Transactions on Computer-Aided Design,* vol. 13, pp. 729-736, Jun. 1994.

[4.9] A. S. Householder, *The Numerical Treatment of a Single Nonlinear Equation.* New York: McGraw-Hill, 1970.

[4.10] X. Huang, V. Raghavan, and R. Rohrer, "AWEsim: A program for the efficient analysis of linear(ized) circuits," in *Tech. Dig. 1990 IEEE/ACM Int. Conf. on Computer-Aided Design,* Nov. 1990.

[4.11] C.-T. Chen, *Linear System Theory and Design,* Third Edition. New York: Oxford University Press, 1999.

[4.12] C. T. Dikmen, M. M. Alaybeyi, S. Topcu, A. Atalar, E. Sezer, M. A. Tan, and R. A. Rohrer, "Piecewise linear asymptotic waveform evaluation for transient simulation of electronic circuits," in *Proc. IEEE Int. Symposium on Circuits and Systems,* pp. 854-857, June 1991.

[4.13] F. J. Alexandro, Jr., "Stable partial Padé approximations for reduced-order transfer functions," *IEEE Transactions on Automatic Control,* vol. 29,1984.

[4.14] M. M. Alaybeyi, J. Y. Lee, and R. A. Rohrer, "Numerical integration and asymptotic waveform evaluation (AWE)," in *Tech. Dig. of the IEEE Int. Conference on Computer-Aided Design,* 1992.

[4.15] E. Chiprout and M. S. Nakhla, "Analysis of interconnect networks using complex frequency hopping (CFH)," *IEEE Transactions on Computer-Aided Design,* vol. 14, no.2, pp. 186-200, Feb. 1995.

[4.16] M. Celik, O. Ocali, M. A. Tan, and A. Atalar, "Pole-zero computation in microwave circuits using multipoint Padé approximation," *IEEE Trans. Circuits Syst. I,* vol. 42, pp. 6-13, Jan. 1995.

CHAPTER 5 *Moment Generation*

In Chapters 2 and 3 we introduced moment-based delay metrics, and in Chapter 4 we presented Asymptotic Waveform Evaluation which uses moments to find Padé approximations of circuit responses. In Chapter 6 we describe Krylov subspace based order reduction methods, such as PRIMA, which are based on Krylov vectors. These Krylov vectors can be viewed as modified moments for numerical conditioning. The common thread through all of these methods are the moments and the efficiency with which they can be calculated.

This chapter summarizes some of the efficient approaches for moment computation in linear circuits. We first describe a path tracing algorithm for tree-like interconnect structures, then a more general explanation of moment generation via the modified nodal analysis (MNA) formulation.

5.1 Calculating Moments in Tree-Like Circuits

In Section 4.2, we demonstrated that moment calculation is performed via successive analyses of a dc equivalent circuit model whereby capacitors are replaced by current sources and inductors are replaced by voltage sources. It follows that the primary factor for guaranteeing efficient moment calculation is to employ a fast dc analysis algorithm. In general, any circuit analysis technique can be used; however, most

interconnect model topologies follow a tree like structure which make them ideal for path tracing methods.

In this section, we first provide a brief introduction to the graph theory and then describe the path-tracing algorithm as employed in RICE [5.1]. The algorithm has an $O(n)$ complexity for RLC trees, and when combined with a circuit compaction method it efficiently handles deviations from strict tree topologies with little degradation in runtime performance.

5.1.1 Circuit Graphs and Incidence Matrix

A graph $G = (V, E)$ consists of a set of objects $V = \{v_1, v_2, \ldots, v_n\}$ called *vertices* or *nodes*, and another set $E = \{e_1, e_2, \ldots, e_b\}$ of *edges* or *branches*, such that each edge e_k is identified with an unordered pair (v_i, v_j) of vertices.

Branches with ends that fall on a node are said to be *incident* at that node. The number of edges incident on a node i is the *degree* of node i. A graph with branches that are oriented is called an *oriented* or a *directed* graph. Electrical circuits are mapped to oriented graphs. Figure 5.1 shows an example of a directed graph with four nodes and six branches. Branch 3 is incident on nodes n1 and n3.

A *loop* is a connected subgraph of a graph at each node of which are incident exactly two branches of the subgraph. Clearly, the algebraic sum of branch voltages around any loop of a circuit is zero (as we will see later, this statement is an alternative way of expressing KVL). In Figure 5.1, {1, 5, 2} is a loop, as is {3, 2, 5, 6}.

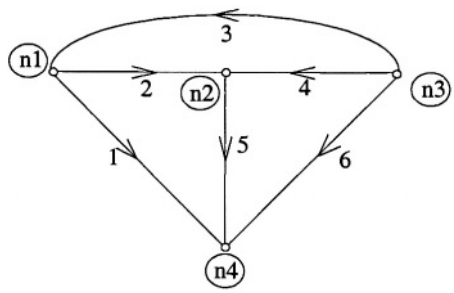


FIGURE 5.1 A directed graph with four nodes and six branches.

A *tree* is a connected subgraph of a graph that contains all the nodes of the graph but no loops. In Figure 5.1, {1, 5, 6} is a tree, as is {3, 2, 5}. {1, 3, 6} is not a tree because it contains a loop, and {1, 5} is not a tree because it does not contain node n3 . Branches comprising the tree are called *tree branches.* The complement of the tree subgraph is called the *cotree.* Branches comprising the cotree are called *cotree links* or just *links.* Note that a graph with (n + 1) nodes has a tree with n branches.

A. Incidence Matrix

For an $(n + 1)$ node, b branch graph without self-loops, the *complete incidence matrix* $\mathbf{A}_a = [a_{ij}]$ is an $(n+1) \times b$ rectangular matrix whose elements have the following values:

- $a_{ij} = 1$ if branch j is incident at node i and oriented away from it.
- $a_{ij} = -1$ if branch j is incident at node i and oriented towards it.
- $a_{ij} = 0$ if branch j is not incident at node i.

For the graph in Figure 5.1,

$$\mathbf{A}_a = \begin{array}{c} \\ n1 \\ n2 \\ n3 \\ n4 \\ \uparrow \\ \text{nodes} \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \leftarrow \text{branches} \\ \begin{bmatrix} 1 & 1 & -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & -1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ -1 & 0 & 0 & 0 & -1 & -1 \end{bmatrix} \end{array} \tag{5.1}$$

Note the following:

- Every column has a +1 and a –1.
- The degree of a node i is the number of nonzeros in row i.
- The sum of all the rows is a row with all zeros, implying that the rows are linearly dependent.
- For an $(n + 1)$ node b branch connected graph, the rank of $\mathbf{A}_a$ is n [5.2].

The *reduced incidence matrix* $\mathbf{A}$ of a graph is obtained by eliminating any row i from its incidence matrix. The rank of $\mathbf{A}$ is n and node i is called the *datum* (or ground) node. In electrical circuits, one node needs to be designated as a reference node

(ground) and the voltage of all other nodes is expressed in terms of this reference potential.

KCL can be conveniently written in terms of the reduced incidence matrix as follows:

$$\mathbf{A}\mathbf{i}_b = \mathbf{0} \tag{5.2}$$

Equation (5.2) assumes that the order of the branches in the columns of $\mathbf{A}$ is the same as the order of branches in the column vector of branch currents, $\mathbf{i}_b$. Clearly, each row represents the branches incident on that node, so when combined with the branch currents, equation (5.2) simply states that the sum of currents leaving (entering) every node is zero.

KVL can also be conveniently written in terms of the reduced incidence matrix as follows:

$$\mathbf{v}_b = \mathbf{A}^T \mathbf{v}_n \tag{5.3}$$

Again, the order of nodes in the vector of node voltages in $\mathbf{v}_n$ is the same as the order of rows in $\mathbf{A}$ and the order of branch voltages in $\mathbf{v}_b$ is the same as the columns of $\mathbf{A}$. The *i*th equation of (5.3) basically asserts that

$$v_{b_i} = v_{from-node_i} - v_{to-node_i} \tag{5.4}$$

5.1.2 The Basic Path-Tracing Algorithm

Before describing the algorithm, we first define our notion of the tree-like topology. If a spanning tree of the circuit model can be constructed that includes all voltage sources, inductors, and resistors and excludes all capacitors and current sources, then it is strictly tree-like. It is obvious that most interconnect circuit models fall into this category.

By definition of a strictly tree-like interconnect circuit, all capacitors must be links, while all resistors and inductors must be tree branches. An example of an RLC interconnect and its spanning tree are shown in Figure 5.2. A spanning tree for a circuit can be efficiently constructed in linear time using a standard algorithm such as that found in [5.3].

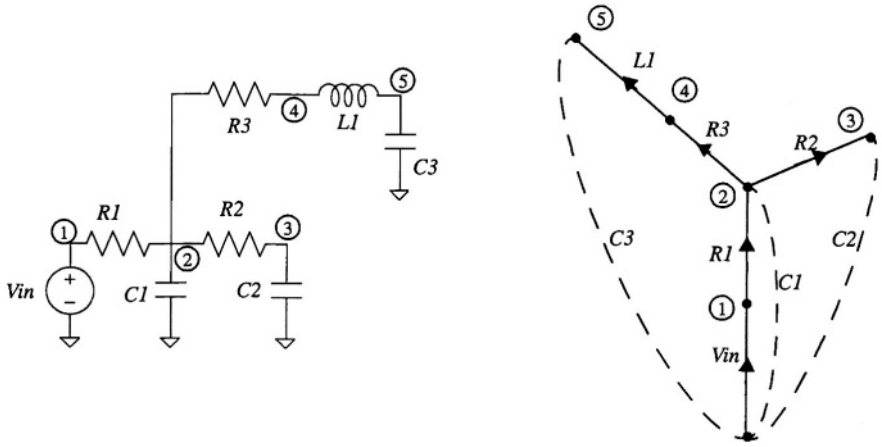


FIGURE 5.2 An RLC interconnect and its graph and spanning tree representation.

Now consider the dc equivalent circuit in Figure 5.3 which will be used to calculate the moments of the circuit shown in Figure 5.2. Once the capacitor-current sources and inductor-voltage sources are assigned their values from previous moment calculations, the tree may be traversed to solve for the currents and voltages of the dc circuit. One complete traversal of the circuit graph is required to compute all branch currents and another traversal is required to yield all node voltages.

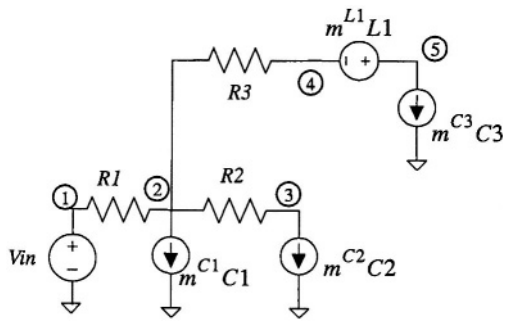


FIGURE 5.3 The dc equivalent circuit for moment generation.

Current calculation: Beginning at any leaf node of the tree, each node is visited by performing a reverse depth-first traversal of the spanning tree. As each node is visited, its incident tree branch and link currents are summed, excluding the current of the tree-branch from the predecessor node. This sum becomes the total current for the tree branch from its predecessor node. The use of reverse depth first traversal guarantees that a node is not visited until the currents for all branches from descendant nodes are known. The process is completed when the root node is encountered. For each inductor tree branch, the resulting current is the new inductor moment to be used in the next moment generation. The currents of tree branch resistors and their corresponding resistance values are then used to compute the resistor branch voltages.

Voltage (moment) calculation: Beginning at the root node, a forward depth first traversal of the tree is performed to visit each node. The voltage (moment) of each node is computed by subtracting the voltage of the predecessor tree branch from the predecessor node voltage. The use of forward traversal guarantees that a node is not visited until the voltage of its parent node is known. The node voltages are then used to compute the voltage of each capacitor, which becomes the new moment for each capacitor used in the next moment generation.

We now present a circuit-theoretic explanation of the path tracing algorithm. First we show that the application of the reverse path trace is equivalent to solving the KCL equations $\mathbf{A}\mathbf{i}_b = \mathbf{0}$, where $\mathbf{A}$ is the reduced incidence matrix described in Subsection 5.1.1.

Consider a circuit graph with n nonground nodes and b branches, and assume that a spanning tree of the graph is constructed. Partitioning the branch current vector $\mathbf{i}_b$, we can write the incidence matrix equation in the form

$$\begin{bmatrix}\mathbf{A}_t & \mathbf{A}_l\end{bmatrix}\begin{bmatrix}\mathbf{i}_t \\ \mathbf{i}_l\end{bmatrix} = \mathbf{0} \tag{5.5}$$

where the subscripts t and l represent the tree branches and links, respectively. Manipulating (5.5) we obtain

$$\mathbf{A}_t\mathbf{i}_t = -\mathbf{A}_l\mathbf{i}_l . \tag{5.6}$$

It is proven that the $n \times n$ matrix $\mathbf{A}_t$ is always nonsingular [5.3], that is, it is invertible. Furthermore, $\mathbf{A}_t$ is always upper-triangular since tree branches are ordered as they would be encountered during a forward traversal of the spanning tree. The

$n \times (b-n)$ matrix $\mathbf{A}_l$ maps the link currents to the nodes, that is, the ith element of the vector $-\mathbf{A}_l\mathbf{i}_l$ is the sum of the currents of the links that are incident to the ith node. Therefore, the tree branch currents can be solved with a simple back substitution. This is, in fact, what is achieved during the reverse path trace. The **A** matrix, however, is not explicitly built. The construction of the circuit graph and the selection of the spanning tree correspond to construction of **A**, and a depth first traversal of the tree is equivalent to ordering **A** .

As an example, consider the dc circuit shown in Figure 5.3, for which (5.6) becomes

$$\begin{bmatrix} 1 & -1 & 0 & 0 & 0 \\ 0 & 1 & -1 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} I_{Vin} \\ I_{R1} \\ I_{R2} \\ I_{R3} \\ I_{L1} \end{bmatrix} = - \begin{bmatrix} 0 \\ I_{C1} \\ I_{C2} \\ 0 \\ I_{C3} \end{bmatrix} \tag{5.7}$$

where we have entered the result of the product $-\mathbf{A}_l\mathbf{i}_l$ directly on the right hand side of the equation. Note that, although obvious from the circuit schematic in this case, the row ordering in (5.7), or more precisely the node ordering in Figure 5.3 which is required for path tracing, would be the result of a depth first search of the tree.

Once the tree branch currents are known, we then calculate the resistor branch voltages. Together with the inductor-voltage source and independent voltage sources they form the tree branch voltages

$$\mathbf{v}_t = \mathbf{R}\mathbf{i}_t + \mathbf{V}_v + \mathbf{V}_{\text{ind}} \tag{5.8}$$

where $\mathbf{V}_v$ and $\mathbf{V}_{\text{ind}} = \mathbf{L}\mathbf{i}_t^{\text{prev}}$ are the vector of independent voltage sources and inductor-voltage sources, respectively, and $\mathbf{i}_t^{\text{prev}}$ is the tree branch currents from the previous moment generation.

In (5.8), **R** and **L** are the resistance and inductance matrices whose nonzero diagonal entries are the branch resistances and inductances, respectively. The nonzero off-diagonal entries in **R** and **L** represent the current-controlled voltage sources and mutual inductances, respectively. For the example circuit of Figure 5.3, **R** and **L** are diagonal, and (5.8) becomes

$$\begin{bmatrix} V_{Vin} \\ V_{R1} \\ V_{R2} \\ V_{R3} \\ V_{L1} \end{bmatrix} = \begin{bmatrix} 0 \\ R1 \cdot I_{R1} \\ R2 \cdot I_{R2} \\ R3 \cdot I_{R3} \\ 0 \end{bmatrix} + \begin{bmatrix} V_{in} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ L1 \cdot I_{L1}^{\text{prev}} \end{bmatrix} \tag{5.9}$$

The application of the forward path trace is equivalent to solving the KVL equations $\mathbf{A}^T\mathbf{v}_n = \mathbf{v}_b$. With partitioning

$$\begin{bmatrix} \mathbf{A}_t^T \\ \mathbf{A}_l^T \end{bmatrix} \mathbf{v}_n = \begin{bmatrix} \mathbf{v}_t \\ \mathbf{v}_l \end{bmatrix}. \tag{5.10}$$

$\mathbf{A}_t^T$ is guaranteed to be lower-triangular, therefore a forward substitution can be applied to solve the node voltages

$$\mathbf{A}_t^T\mathbf{v}_n = \mathbf{v}_t. \tag{5.11}$$

For example, for the circuit in Figure 5.3, (5.11) becomes

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \\ V_5 \end{bmatrix} = \begin{bmatrix} V_{Vin} \\ V_{R1} \\ V_{R2} \\ V_{R3} \\ V_{L1} \end{bmatrix} \tag{5.12}$$

The link voltages are calculated from the node voltages as

$$\mathbf{v}_l = \mathbf{A}_l^T\mathbf{v}_n \tag{5.13}$$

We then update the capacitor-current source values for the next generation of moments

$$\mathbf{i}_l^{\text{next}} = \mathbf{I}_i + \mathbf{C}\mathbf{v}_l \tag{5.14}$$

where $\mathbf{I}_i$ is the vector of independent current sources and $\mathbf{C}$ is the diagonal capacitance matrix. Note that $\mathbf{C}$ is always diagonal even in the presence of coupling capacitances.

We should point out that a dc circuit need only be (actually) path traced one time regardless of the number of moment generations required. Since neither the circuit graph nor spanning tree changes between each generation, the nodes will be visited in the same order during each moment generation. Moreover, the amount of information at each node (links and tree branches) does not change between generations. This suggests that instead of executing an actual traversal of the graph and tree for each analysis, the traversal can be performed only one time and we can *vectorize* the tree branch and link elements in order of a depth-first traversal of the spanning tree.

5.1.3 Extension for Resistor Loops

A particularly important non-tree circuit topology that often appears for interconnect structures is one containing resistor loops or links (*R*-loops or *R*-links). For example, power/ground distribution circuits are often implemented as a grid. Similarly, some clock lines are routed as a mesh.

Resistors that cause loops in the graph are classified as resistor links during the construction of the spanning tree. The resistor links are undesirable since their currents are not known *a priori.* This means we cannot use (5.6) to calculate the tree-branch currents.

A. Branch Tearing

One way to handle link resistors in the AWE dc circuit is by branch tearing or Kron's method [5.4] [5.2]. Consider a circuit with a single link resistor as shown in Figure 5.4. Assume that the circuit is solved to obtain the open circuit voltage, v_{oc}, across the open link resistor. From Kron's method, the current that would flow through the link, were it not opened, is

$$I_R = \frac{v_{oc}}{R_{link} + R_{th}} \tag{5.15}$$

where R_{th} is the Thevenin resistance and R_{link} is the link resistance. R_{th} is obtained by summing the resistances of the tree branches that form a fundamental loop with the link resistor. For the example of Figure 5.4 it is equivalent to $R_1 + R_2$.

The link resistor is then replaced by a current source of value I_R, thus restoring the tree structure of the dc equivalent circuit and allowing the circuit to be solved by path tracing. Note that only the tree branches that are in the fundamental loop are affected by I_R. In the case where there are multiple loops, if the loops are isolated, we can solve the R-link currents by applying the above method to each loop separately.

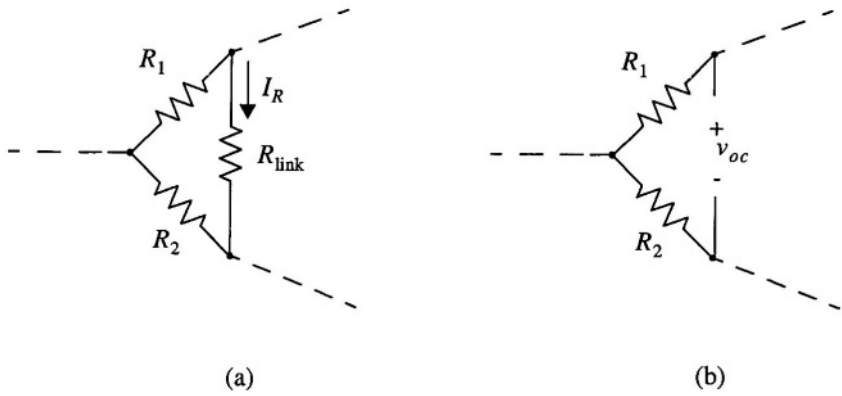


FIGURE 5.4 (a) A tree with a link resistor and (b) the open link circuit to find the Thevenin voltage.

If the loops are not isolated, however, the problem becomes more complicated, but the approach is still basically the same. We need to calculate the current value for each R-link that would flow were they not opened. Therefore, we need to extend (5.15) to the general case of m R-links. The Kron method also supports this case, where I_R and v_{oc} become vectors of size m and $(R_{\text{link}} + R_{\text{th}})$ is replaced by $\mathbf{Z}$ as follows:

$$\mathbf{I}_R = \mathbf{Z}^{-1}\mathbf{v}_{oc} \tag{5.16}$$

The impedance matrix is given by

$$\mathbf{Z} = \mathbf{F}\mathbf{R}_t\mathbf{F}^T + \mathbf{R}_l \tag{5.17}$$

where $\mathbf{F}$ is the $m \times N$ fundamental loop/cutset matrix [5.2], $\mathbf{R}_t$ is the $N \times N$ diagonal matrix of the tree-branch resistors, and $\mathbf{R}_l$ is the $m \times m$ diagonal matrix of link resistors, and, N and m are, respectively, the numbers of tree branch and link resistors.

The vector $\mathbf{v}_{oc}$ is the torn or open-circuit voltages across the torn branches, which are easily obtained by an initial path tracing.

Solving the system for the *R*-link currents, $\mathbf{I}_R$, requires the inversion of **Z**, which is $m \times m$ and symmetric for resistive circuits but can be very dense. This, of course, depends on **F**, which in turn depends on the spanning tree that is selected for the circuit graph. Once again, we emphasize that the matrix **F** is not explicitly constructed. Instead, a path tracing algorithm can be used to build **Z** directly from the circuit graph [5.1]. Another path tracing yields the open-circuit voltages (i.e., with the *R*-links not present), and solving (5.16) yields the *R*-link currents. Finally, these currents are substituted for the torn branches and the circuit is path traced one last time to obtain the final circuit solution.

Both the construction of **Z** and its inversion can be expensive. Empirical results suggest that this method is suitable for circuits with few resistor loops ($m < 100$), but tends to be overwhelming for circuits containing a large number of loops [5.1].

B. Solving by Circuit Compaction

A more efficient way to handle link resistors in the AWE dc circuit is by circuit compaction [5.5] [5.1]. The goal of this method is to formulate a smaller equivalent circuit that may be solved to obtain the resistor-link currents. The circuit is reformulated such that the original resistor loops are left intact, but Norton-equivalent subcircuits are substituted for remaining sections of the circuit between the loops. These Norton equivalents are easily constructed during the initial path tracing with *R*-loops opened (*R*-links removed).

The compaction scheme replaces all long tree branch sections between successive R-loops with a super tree-branch (STB). This is demonstrated in Figure 5.5. The STB resistance is simply the sum of all resistances between the ends of the STB. The Thevenin voltage is the voltage difference across the STB with all *R*-links opened. This is illustrated in the fourth sub-figure in Figure 5.5 where the Thevenin voltage is $V(x) - V(y)$ and the Thevenin resistance is $R_1 + R_2$. The total downstream current through the STB is the sum of the capacitor current sources, I_C, and the unknown *R*-link currents. The capacitor current will be known after the initial path tracing.

A Norton equivalent is used instead of a Thevenin model to facilitate the use of nodal analysis to solve the compact circuit. An initial path trace is required to calculate the Norton resistances. The starting and ending nodes of the STB are retained in the compact circuit equivalent, while the intermediate nodes are deleted. The folding up of the

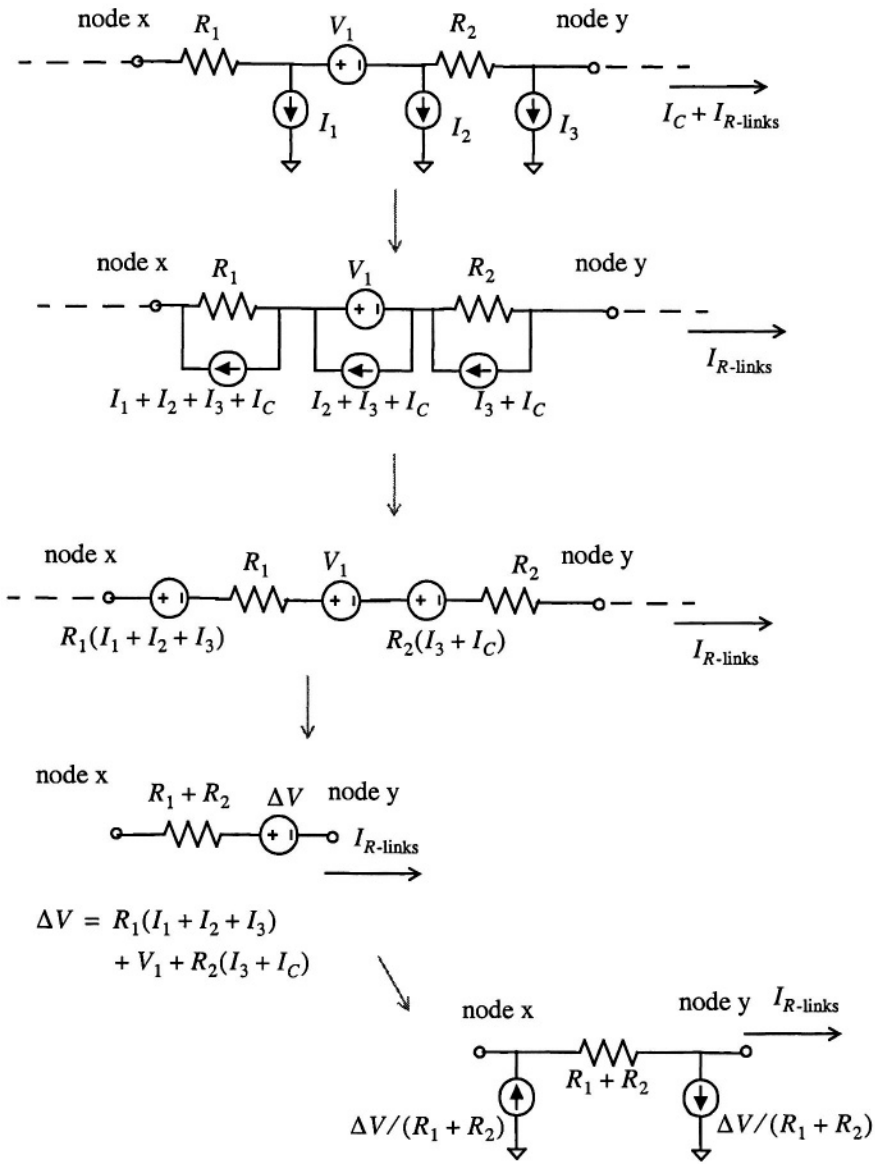


FIGURE 5.5 Compaction of a tree-branch segment into a super tree-branch (STB).

current sources shown in Figure 5.5 is allowed by the source transformation [5.6] and occurs naturally during the path trace. Conceptually, the values of the current sources are modified so that the net current for each branch is unchanged. Figure 5.5 is used only to illustrate the concept. We do not actually modify the current values, but we jump immediately to the model in the last sub-figure, which is easily obtained during path tracing.

What may not be immediately obvious is how the compact circuit can be efficiently formulated. The first step is to decide which nodes in the main circuit must be retained in the compact circuit. The most obvious nodes in the circuit that must be retained are those with one or more incident resistor links. Other nodes that must be retained are those that root two or more subtrees, each containing at least one resistor link. This requirement guarantees that the mutual effects of resistor links on one another are the same in the compact circuit as in the main circuit. A node that roots two or more subtrees, where only one subtree contains R-links, need not be retained since the effects of these subtrees will be reflected in the Norton currents. Likewise, a node that roots two or mores subtrees with no resistor links may be discarded. After determining the nodes in the compact circuit, the Norton resistances may be found by summing all resistor values between mapped nodes. This is accomplished with a single depth first traversal of the spanning tree.

An example of circuit compaction is illustrated in Figure 5.6. It is assumed that Figure 5.6(a) is the AWE dc circuit obtained from an RLC circuit. It has two *R*-links: R_4 and R_8. Note that the compact circuit contains only four of the original 11 nodes. Node 5 was retained in the compact circuit since it roots two subtrees that contain *R*-links. The topology of the compact circuit does not vary between moment generations, but the currents of the Norton sources, shown in Figure 5.6(b), must be recomputed between each generation. This is accomplished with a single path trace.

C. Formulating the Compact Circuit Equations

The solution of the compact circuit is obtained by formulating and solving the node voltage equations

$$\mathbf{G}_{cc}\mathbf{v}_{cc} = \mathbf{i}_{cc} \tag{5.18}$$

where $\mathbf{G}_{cc}$ is the $n \times n$ node-conductance matrix for the compact circuit, $\mathbf{v}_{cc}$ is the vector of compact node voltages, $\mathbf{i}_{cc}$ is the vector of the sum of Norton currents at each node, and n is the number of nodes in the compact circuit.

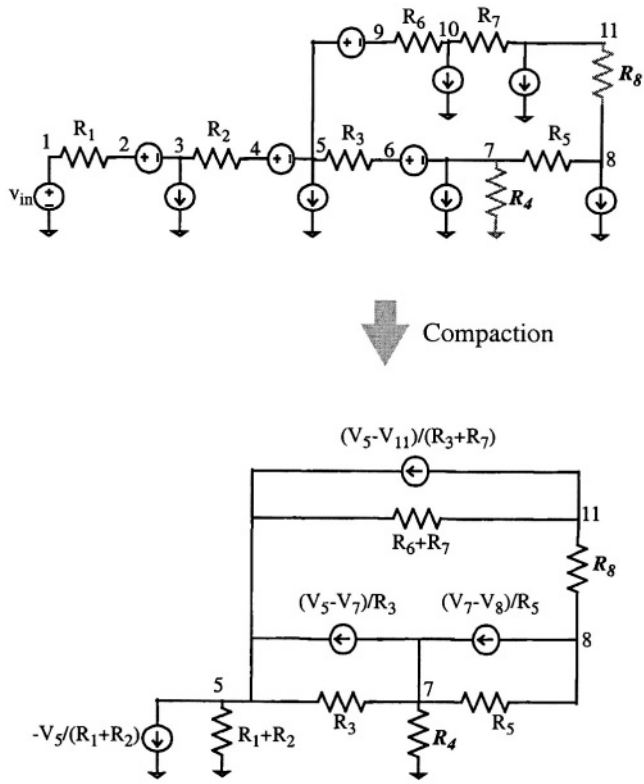


FIGURE 5.6 An AWE dc circuit before and after compaction.

In (5.18), the matrix $\mathbf{G}_{cc}$ is symmetric and positive-definite, thus allowing a Cholesky factorization [5.7] to be performed. This accomplishes the decomposition

$$\mathbf{G}_{cc} = \mathbf{L}\mathbf{L}^{T} \tag{5.19}$$

which is inherently more efficient than LU decomposition. Cholesky factorization is more efficient than LU for two reasons. First, due to symmetry, only half of the usual number of floating-point operations are required. Secondly, no pivoting is required, thus allowing the use of very efficient matrix storage and ordering algorithms. For example, RICE uses the reverse Cuthill-McGee technique for ordering the equations and a vector-based format for storing and manipulating the matrix [5.7].

D. Obtaining the Final Solution

The formulation of the compact circuit and associated equations and decomposition is a one-time cost associated with circuit setup and is performed before any moments are computed. To generate a set of moments, the circuit is initially solved via a path-tracing as R-links opened. Next, the Norton currents are calculated, summed to the right hand side of (5.18), and then a Cholesky forward and back substitution yields the compact circuit node voltages. The node voltages in the compact circuit correspond to voltages in the original circuit, so they may be used to directly compute the current for each *R*-loop. These currents are substituted into the original circuit and a second path tracing is performed to produce the final dc solution.

5.1.4 Extension for Floating Nodes and Inductor Loops

There are some interconnect models in which floating nodes or loops of inductors may be present. A floating node has no dc path to ground, thus causing a pure cutset of capacitors in the circuit graph. A loop of inductors is caused when a closed path of inductors is specified with no intervening resistance or capacitance. These problems can be solved by the application of charge and flux conversation, respectively [5.8]. Also, it is desirable to solve these types of circuits in a manner that is decoupled from the solution of resistor loops, thus preserving the efficiency of the *R*-loop solution.

A. Floating Nodes

For an interconnect model, a floating node suggests that two interconnect nets are capacitively coupled and one net has no active driver. This may be true for a line that is driven by a three-state device in its high-impedance state or for an interconnect circuit model that is constructed to measure clock feedthrough.

Consider the circuit shown in Figure 5.7(a). Node 4 is a floating node since every path from it to ground contains a series capacitor. The dc equivalent AWE circuit is shown in Figure 5.7(b) where all the capacitors are replaced by capacitor-current sources. As apparent from the figure, the circuit contains an illegal cutset among I_2, I_3, and I_4 which causes a problem. First, if the sum of the currents is not zero, then KCL is violated at Node 4. Therefore, we need a consistent equation,

$$I_2 = I_3 + I_4 \tag{5.20}$$

However, even if the KCL were satisfied, it would be a redundant equation and the voltage values at Node 4 and Node 5 would be undefined. Thus we need to supplement the circuit equations with an additional constraint to resolve the voltages at these nodes but we still must keep current source cutset equation as consistent and redundant.

Since charge conversation must hold for this circuit just as it must be for all dc circuits, we can use the following equation to supplement the circuit equations [5.8]:

$$C_2(V_3 - V_4) - C_3V_4 - C_4V_5 = 0. \tag{5.21}$$

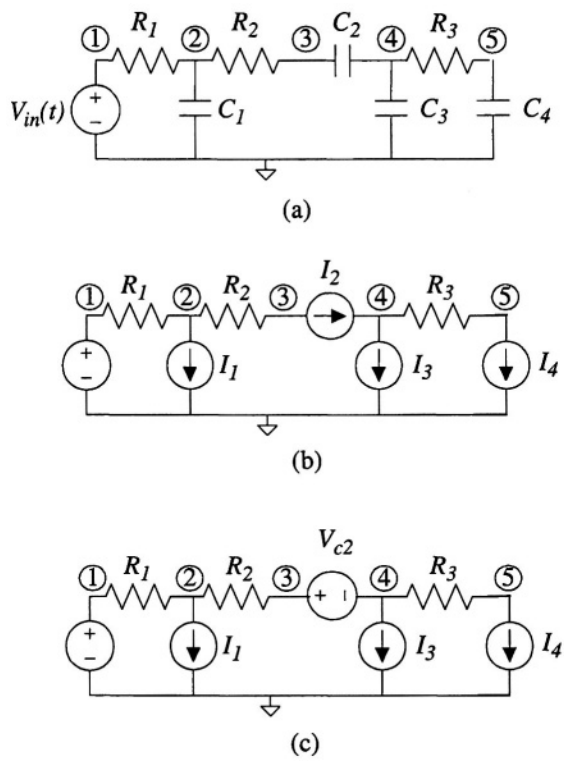


FIGURE 5.7 (a) Interconnect models containing floating nodes, (b) its dc equivalent AWE circuit, and (c) modeling the tree-capacitor C_2 as a voltage controlled voltage source.

Now let us remember that when solving for the $(k+1)$th set of voltage moments, the current source corresponding to capacitor C_i is set equal to the product of C_i and capacitance voltage from the kth set. Therefore using the charge conversation equation to calculate the kth set of moments guarantees that the equations for the $(k+1)$th set of moments will be consistent.

In nodal analysis the charge conversation equations are simply substituted for the redundant current source equations as described in [5.2]. We now discuss how we can use charge conversation equations to resolve the floating node situation in path tracing.

In path tracing, when an illegal cutset of capacitors is formed, it causes one of the capacitors to be replaced in the spanning tree. Then, one charge conversation equation is formulated for each tree-capacitor. Thus, for a circuit with m tree-capacitors, a $m \times m$ linear system of equations results. For example, consider the example in Figure 5.7 where C_2 is selected as the tree-capacitor. In the AWE circuit, it is replaced by a (voltage-controlled) voltage source and the voltage across it can be resolved by solving the following set of equations:

$$\begin{aligned} C_2V_{C_2} - C_3V_{C_3} - C_4V_{C_4} &= 0 \\ V_{C_2} + V_{C_3} &= V_1 - V_{R_1} - V_{R_2} \\ V_{C_2} + V_{C_4} &= V_1 - V_{R_1} - V_{R_2} - V_{R_3} \end{aligned} \tag{5.22}$$

Note that the first equation is the charge conversation equation given in (5.21). The last two KVL equations are easily eliminated using substitution and a single equation is obtained for the tree capacitor voltage:

$$(C_2 + C_3 + C_4)V_{C_2} = C_3\Delta V_{C_2, C_3} + C_4\Delta V_{C_2, C_4} \tag{5.23}$$

where $\Delta V_{C_2, C_3}$ and $\Delta V_{C_2, C_4}$ are, respectively, the right hand sides of the second and third equations in (5.22), and they are calculated during path tracing. By combining the equations for m tree-capacitors we obtain an $m \times m$ system equations. Note that some of these equations can be coupled.

This system is formulated and LU-factored prior to computing any moments. Computation of moments requires two path traces. In the initial trace, the branch currents are calculated and their values do not depend on the tree-capacitor voltages. A subset of the results of this path trace are used to setup the right hand side of the m linear equations. Next, these equations are solved via a forward and back substitution step,

resulting in the actual tree-capacitor voltages. Finally, these voltages are substituted in the place of the zero-values and the circuit is path traced a second time to get the final dc solution.

B. Loops of Inductors

Another situation that can occur in certain interconnect models is inductance loops. This type of situation arises, for example, in the modeling of backplane or pc-board interconnections where two devices drive a line simultaneously and the metal between them is modeled as a lossless inductive line.

The solution for loops of inductors is simply the dual of the floating node problem. A loop of inductors will cause one of the inductors to be excluded from the spanning tree and create an inductor link. Since inductors in AWE are modeled as voltage sources, this again presents a problem. There is no well defined dc solution for a loop of voltage sources. In this case, flux conversation must be applied to resolve the unknown current of each inductor link.

C. Mixing Loops of Resistors, Loops of Inductors, and Floating Nodes

If a circuit contains a combination of the three types of troublesome elements, then they must be solved in a specific order to guarantee a valid solution. Link inductors must be solved first, followed by tree capacitors, and then link resistors. A loop of inductors can only contain other inductors and a cutset of capacitors may contain only other capacitors, so these do not affect each other and are not affected by changes of current in loop resistors.

5.2 Calculating Moments Using MNA

Tree structured interconnect circuits can be analyzed very efficiently using path tracing. With some extensions, such as compaction and factorization, path tracing still works very well when the circuit model contains non-tree elements such as resistor links. However, as the circuit becomes more and more non-tree structured, the path tracing method loses its efficiency and in the limit it becomes equivalent to a nodal analysis. In this section, we first provide a formal derivation of modified nodal analysis (MNA) formulation and then describe moment generation in terms MNA formulation.

5.2.1 Modified Nodal Analysis (MNA) Formulation

MNA [5.9], which is an extension of the nodal analysis method, can handle all types of circuit elements, and is well suited to computer applications. Although there exist more efficient circuit formulation techniques for specific circuit topologies, MNA is very easy to implement and equally suitable for both frequency and time-domain analyses. Because of these nice properties it constitutes the standard formulation method for computer-based circuit analysis [5.10] and is used in general purpose circuit simulators such as SPICE.

Consider a linear circuit which consists of resistors, inductors, capacitors, and independent voltage and current sources. The current and voltage vectors of the circuit can be partitioned such that the matrix KCL equation, defined in (5.2), can be written in the form

$$\begin{bmatrix}\mathbf{A}_g & \mathbf{A}_c & \mathbf{A}_l & \mathbf{A}_v & \mathbf{A}_i\end{bmatrix}\begin{bmatrix}\mathbf{I}_g\\ \mathbf{I}_c\\ \mathbf{I}_l\\ \mathbf{I}_v\\ \mathbf{I}_i\end{bmatrix} = \mathbf{0} \tag{5.24}$$

where the subscripts g, c, l, v, and i, respectively, represent branches containing resistors, capacitors, inductors, voltage sources, and current sources.

With this partitioning the KVL equations become

$$\mathbf{V}_g = \mathbf{A}_g^T\mathbf{V}_n \tag{5.25}$$

$$\mathbf{V}_c = \mathbf{A}_c^T\mathbf{V}_n \tag{5.26}$$

$$\mathbf{V}_l = \mathbf{A}_l^T\mathbf{V}_n \tag{5.27}$$

$$\mathbf{V}_v = \mathbf{A}_v^T\mathbf{V}_n \tag{5.28}$$

$$\mathbf{V}_i = \mathbf{A}_i^T\mathbf{V}_n \tag{5.29}$$

where $\mathbf{V}_n$ is the vector of all node voltages.

The branch constitutive relations for the first three partitions are as follows:

$$G\mathbf{V}_g = \mathbf{I}_g \tag{5.30}$$

$$sC\mathbf{V}_c = \mathbf{I}_c \tag{5.31}$$

$$sL\mathbf{I}_l = \mathbf{V}_l \tag{5.32}$$

Here G, C, and L are matrices whose elements are the conductance, capacitance, and inductance values of each element, respectively. The equations from (5.24) to (5.32) are the required set of circuit equations to solve the circuit. Modified nodal analysis (MNA) formulation combines these equations in a more compact form by eliminating as many unknowns as possible. Rewriting (5.24) and substituting (5.30) and (5.31) for $\mathbf{I}_g$ and $\mathbf{I}_c$ we obtain

$$\mathbf{A}_g G\mathbf{V}_g + s\mathbf{A}_c C\mathbf{V}_c + \mathbf{A}_l\mathbf{I}_l + \mathbf{A}_v\mathbf{I}_v = -\mathbf{A}_i\mathbf{I}_i \tag{5.33}$$

Branch voltages $\mathbf{V}_g$ and $\mathbf{V}_c$ are eliminated by substituting (5.25) and (5.26):

$$\mathbf{A}_g G\mathbf{A}_g^T\mathbf{V}_n + s\mathbf{A}_c C\mathbf{A}_c^T\mathbf{V}_n + \mathbf{A}_l\mathbf{I}_l + \mathbf{A}_v\mathbf{I}_v = -\mathbf{A}_i\mathbf{I}_i \tag{5.34}$$

Substituting (5.27) into (5.32),

$$\mathbf{A}_l^T\mathbf{V}_n - sL\mathbf{I}_l = \mathbf{0} \tag{5.35}$$

Combining (5.34), (5.35), and (5.28), we obtain the MNA formulation

$$\begin{bmatrix} \mathbf{A}_g G\mathbf{A}_g^T + s\mathbf{A}_c C\mathbf{A}_c^T & \mathbf{A}_l & \mathbf{A}_v \\ \mathbf{A}_l^T & -sL & \mathbf{0} \\ \mathbf{A}_v^T & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{V}_n \\ \mathbf{I}_l \\ \mathbf{I}_v \end{bmatrix} = \begin{bmatrix} -\mathbf{A}_i\mathbf{I}_i \\ \mathbf{0} \\ \mathbf{V}_v \end{bmatrix} \tag{5.36}$$

Denoting

$$\begin{bmatrix} \mathbf{A}_g G \mathbf{A}_g^T & \mathbf{A}_l & \mathbf{A}_v \\ \mathbf{A}_l^T & 0 & 0 \\ \mathbf{A}_v^T & 0 & 0 \end{bmatrix} = \mathbf{G} \quad \text{and} \quad \begin{bmatrix} \mathbf{A}_c C \mathbf{A}_c^T & 0 & 0 \\ 0 & -L & 0 \\ 0 & 0 & 0 \end{bmatrix} = \mathbf{C} \tag{5.37}$$

the MNA matrix equation can be represented in the compact form

$$(\mathbf{G} + s\mathbf{C})\mathbf{X} = \mathbf{b} \tag{5.38}$$

where $\mathbf{X}$ is the solution vector containing node voltages appended by inductance and voltage source currents,

$$\mathbf{X} = \begin{bmatrix} \mathbf{V}_n & \mathbf{I}_l & \mathbf{I}_v \end{bmatrix}^T \tag{5.39}$$

and $\mathbf{b}$ is the excitation vector,

$$\mathbf{b} = \begin{bmatrix} -\mathbf{A}_i \mathbf{I}_i & 0 & \mathbf{V}_v \end{bmatrix}^T \tag{5.40}$$

Although we have showed the derivation of MNA formulation only for RLC circuits, any type of linear(ized) element can be inserted into MNA formulation and the form in (5.38) is still retained. Moreover, its implementation actually does not require any graphs or incidence matrices. MNA matrices can be built by inspection on a branch-by-branch basis by stamping the elements [5.10].

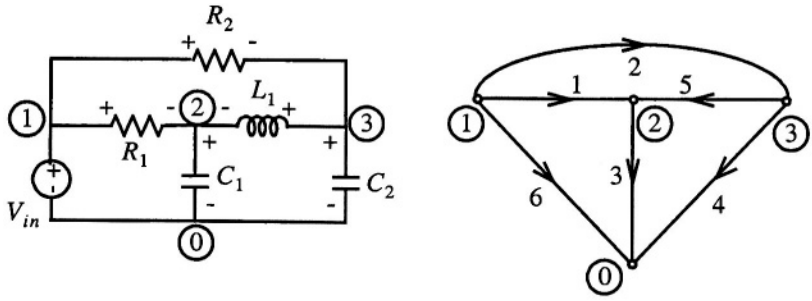


FIGURE 5.8 A circuit and its graph.

Consider the example circuit shown in Figure 5.8. The branch numbering is modified so that the partitions in the branch vector are ordered as defined in (5.24). From Figure 5.8 and (5.24), we have

$$\mathbf{I}_g = \begin{bmatrix} I_1 \\ I_2 \end{bmatrix} \quad \mathbf{I}_c = \begin{bmatrix} I_3 \\ I_4 \end{bmatrix} \quad \mathbf{I}_l = I_5 \quad \mathbf{I}_v = I_6 \tag{5.41}$$

$$\mathbf{A}_g = \begin{bmatrix} 1 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} \quad \mathbf{A}_c = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \mathbf{A}_l = \begin{bmatrix} 0 \\ -1 \\ 1 \end{bmatrix} \quad \mathbf{A}_v = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{5.42}$$

The matrices in branch constitutive relations are

$$G = \begin{bmatrix} G_1 & 0 \\ 0 & G_2 \end{bmatrix} \quad C = \begin{bmatrix} C_1 & 0 \\ 0 & C_2 \end{bmatrix} \quad L = L_1 \tag{5.43}$$

where $G_1 = 1/R_1$ and $G_2 = 1/R_2$.

The MNA matrices and vectors become

$$\mathbf{X} = \begin{bmatrix} V_{n,1} & V_{n,2} & V_{n,3} & I_4 & I_5 \end{bmatrix}^T \quad \mathbf{b} = \begin{bmatrix} 0 & 0 & 0 & 0 & V_{in} \end{bmatrix}^T \tag{5.44}$$

$$\mathbf{G} = \begin{bmatrix} G_1 + G_2 & -G_1 & -G_2 & 0 & 1 \\ -G_1 & G_1 & 0 & 1 & 0 \\ -G_2 & 0 & G_2 & -1 & 0 \\ 0 & 1 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{5.45}$$

$$\mathbf{C} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & C_1 & 0 & 0 & 0 \\ 0 & 0 & C_2 & 0 & 0 \\ 0 & 0 & 0 & -L_1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{5.46}$$

5.2.2 Calculating Moments Using MNA

Consider a linear circuit and its MNA formulation of the form

$$(\mathbf{G} + s\mathbf{C})\mathbf{X} = \mathbf{b} \tag{5.47}$$

where **G**, **C**, **X**, and **b** are defined in (5.37), (5.39), and (5.40).

Now assume a moment expansion for the solution vector **X**:

$$\mathbf{X}(s) = \mathbf{x}_0 + \mathbf{x}_1 s + \mathbf{x}_2 s^2 + \ldots \tag{5.48}$$

Inserting (5.48) into (5.47), we obtain

$$(\mathbf{G} + s\mathbf{C})(\mathbf{x}_0 + \mathbf{x}_1 s + \mathbf{x}_2 s^2 + \ldots) = \mathbf{b} \tag{5.49}$$

or

$$\mathbf{G}\mathbf{x}_0 + s(\mathbf{G}\mathbf{x}_1 + \mathbf{C}\mathbf{x}_0) + s^2(\mathbf{G}\mathbf{x}_2 + \mathbf{C}\mathbf{x}_1) + \ldots = \mathbf{b} \tag{5.50}$$

The above equation implies that the moment vectors can be calculated recursively as

$$\mathbf{G}\mathbf{x}_k = -\mathbf{C}\mathbf{x}_{k-1} \tag{5.51}$$

with

$$\mathbf{G}\mathbf{x}_0 = \mathbf{b} \tag{5.52}$$

Equation (5.52) is the dc solution of the circuit. For higher order moments the same matrix equation is solved with a different input vector. To solve (5.52), first an LU decomposition is performed for the matrix **G**,

$$\mathbf{G} = \mathbf{L}\mathbf{U} \tag{5.53}$$

where **L** and **U** are, respectively, lower and upper triangular matrices. Then using a simple forward substitution an auxiliary vector is obtained:

$$\mathbf{L}\mathbf{z} = \mathbf{b} \tag{5.54}$$

Then the solution vector is obtained with a back substitution as

$$\mathbf{U}\mathbf{x}_0 = \mathbf{z} \tag{5.55}$$

Once the LU factors of the matrix **G** is obtained for the first moment, the higher order moments can be calculated with simple forward and back substitutions.

Using sparse techniques it has been shown that the cost of finding LU factors is $O(n^{1.4-1.7})$ for matrices that typically arise in circuit simulation, where n is the matrix dimension [5.11]. For typical interconnect circuits, the matrix **G** is known to be extremely sparse having only few nonzero elements at each row. It has been reported that by suitably tuning the sparse matrix techniques for tree type interconnect circuits, the cost of LU decomposition can become very close to $O(n)$ [5.12].

The choice of path tracing or matrix factorization depends on the circuit topology and the number of resistor links. If it is known that **G** is symmetric positive definite, which is the case in RC circuits with current source excitations, then Cholesky factorization can be used for further speedup over LU decomposition. In general, however, path tracing has less computational overhead when compared with either LU or Cholesky factorization.

One situation for which matrix factorization is preferred over path tracing is when there are numerous grounded resistors. When frequency shifting is applied to improve the stability and accuracy of moment matching, the dc equivalent moment-generation circuit represents capacitors by Norton equivalents, thereby establishing a resistor from every node to ground for a typical interconnect circuit.

A. Calculating Frequency-Shifted Moments

We introduce a new Laplace variable s', such as

$$s' = s - \alpha \tag{5.56}$$

where α is the shift amount. Then, the MNA equations become

$$(\mathbf{G} + (s' + \alpha)\mathbf{C})\mathbf{X} = \mathbf{b} \tag{5.57}$$

or

$$((\mathbf{G} + \alpha\mathbf{C}) + s'\mathbf{C})\mathbf{X} = \mathbf{b} \tag{5.58}$$

The moments around $s' = 0$ $(s = \alpha)$ can be calculated in a similar manner to standard AWE. The only difference is that now the matrix $\mathbf{G}$ in (5.51) and (5.52) is replaced by $\mathbf{G} + \alpha\mathbf{C}$. However, since $\mathbf{C}$ may have nonzero entries where the entries of $\mathbf{G}$ were zero, the sparsity of the matrix is often destroyed by frequency shifting. This may diminish the computational efficiency of the LU factorization.

On the other hand, frequency shifting has one advantage; it breaks the capacitance cutsets and inductance loops, which would otherwise have rendered the matrix $\mathbf{G}$ singular.

5.3 Calculating Moments in Transmission Line Circuits

So far, we have explained how to calculate moments for linear lumped circuits. We now describe a method for moment generation for general linear circuits; namely those which can contain distributed elements. Consider a circuit and its system equation

$$\mathbf{T}(s)\mathbf{X} = \mathbf{b} \tag{5.59}$$

with an output function

$$H(s) = \mathbf{d}^T\mathbf{X}. \tag{5.60}$$

If the circuit has transmission lines and retarded coupling elements in addition to lumped components then its system matrix includes transcendental functions and is regarded as an infinite-dimensional system. The circuit functions for an infinite-dimensional systems have infinite number of poles and they cannot be expressed as a ratio of two polynomials of finite degree. However, they can be approximated with a finite order system using AWE. For this we first need the moments of $H(s)$.

Consider the Taylor series expansion of the system matrix

$$\mathbf{T}(s) = \mathbf{T}_0 + s\mathbf{T}_1 + s^2\mathbf{T}_2 + \ldots \tag{5.61}$$

We wish to compute the moments of the solution vector $\mathbf{X}$ so that we can obtain the moments of $H(s)$. Inserting (5.61) into (5.59) and expanding $\mathbf{X}$, we get

$$(\mathbf{T}_0 + s\mathbf{T}_1 + s^2\mathbf{T}_2 + \ldots)(\mathbf{x}_0 + \mathbf{x}_1 s + \mathbf{x}_2 s^2 + \ldots) = \mathbf{b}. \tag{5.62}$$

Then it is easy to show that the moment vectors can be computed recursively as

$$\begin{aligned} \mathbf{T}_0\mathbf{x}_0 &= \mathbf{b} \\ \mathbf{T}_0\mathbf{x}_k &= -\sum_{i=1}^{k} \mathbf{T}_i\mathbf{x}_{k-i} \qquad k>0 \end{aligned} \tag{5.63}$$

Therefore, the moment computation needs only one matrix decomposition similar to the lumped circuits.

Once the moments of $H(s)$ are computed, a reduced order model, $\hat{H}(s)$, can be found by applying moment matching as described in the previous chapter.

5.3.1 Transmission Lines

A general way to include a transmission line system into the MNA formulation is to treat it as a linear N-port system with a frequency-domain terminal equation in the form

$$\mathbf{A}(s)\mathbf{V}_t + \mathbf{B}(s)\mathbf{I}_t = \mathbf{0} \tag{5.64}$$

where $\mathbf{V}_t$ and $\mathbf{I}_t$ are the s-domain terminal voltages and currents, respectively. The matrices $\mathbf{A}$ and $\mathbf{B}$ are described in terms of the per-unit-length line parameters and are usually exponential type functions of s. With this approach, arbitrary number of N-port transmission lines can be combined with linear lumped elements to obtain a system matrix equation in the form of (5.59) for the entire circuit.

Obviously, the most important step in the order reduction of linear circuits containing transmission lines is the determination of the coefficients given in (5.61) which requires the derivatives of $\mathbf{A}$ and $\mathbf{B}$. For this purpose, a method, which is called matrix exponential moment method, is proposed in [5.13]. It is generalized for dispersive transmission lines in [5.14].

Instead of applying order reduction to the entire circuit which may contain transmission lines, we can also treat each transmission line system separately. Such an approach, which combines AWE with the method of characteristics, is proposed in

[5.15]. The "pure delay" factors are computed exactly and extracted from the propagation functions of the lines. The remainder responses can be viewed as containing the attenuation and dispersion behavior of the propagation responses. A conventional moment matching is used to accurately approximate the propagation response without delay.

5.4 Summary

In this chapter we described efficient methods for generating moments for linear circuits. Among these methods the path-tracing algorithm achieves the optimum efficiency for RLC tree-like interconnect structures, and for many interconnect topologies that contain resistor loops. The MNA formulation, in addition to being general and very simple to implement, can provide excellent runtime efficiency when combined with sparse matrix techniques and special ordering algorithms.

References

[5.1] C. L. Ratzlaff and L. T. Pillage, "RICE: Rapid interconnect circuit evaluation using AWE," *IEEE Transactions on Computer-Aided Design,* pp. 763-776, Jun. 1994.

[5.2] L. T. Pillage, R. A. Rohrer, and C. Visweswariah, *Electronic Circuit and System Simulation Methods.* New York: McGraw-Hill, Inc., 1995.

[5.3] N. Deo, *Graph Theory with Applications to Engineering and Computer Science.* New York: Prentice-Hall, 1974.

[5.4] G. Kron, *Tensor Analysis of Networks.* New York: Wiley, 1939.

[5.5] D. Stark and M. Horowitz, "Techniques for calculating current and voltages in VLSI power supply networks," *IEEE Transactions on Computer-Aided Design,* vol. 9, no. 2, pp. 126-132, Feb. 1990.

[5.6] C. A. Desoer and E. S. Kuh, *Basic Circuit Theory.* New York: McGraw-Hill, 1969.

[5.7] S. Pissanetzky, *Sparse Matrix Technology.* London: Academic Press, 1984.

[5.8] L. T. Pillage, X. Huang, and R. A. Rohrer, "Asymptotic waveform evaluation for circuits containing floating nodes," in *Proc. IEEE Int. Symposium on Circuits and Systems,* pp. 613-616, May 1990.

[5.9] C. W. Ho, A. E. Ruehli, and P. A. Brennan, "The modified nodal approach to network analysis," *IEEE Transactions on Circuits and Systems,* vol. CAS-22, pp. 504-509, June 1975.

[5.10] J. Vlach and K. Singhal, *Computer Methods for Circuit Analysis and Design,* Second Edition. New York: Van Nostrand Reinhold, 1994.

[5.11] K. S. Kundert, *Sparse matrix techniques,* Chapter on *Circuit analysis, simulation and design, Advances in CAD for VLSI,* A.E. Ruehli (ed.) vol. 3, part 1, pp. 281-324, Amsterdam: Elsevier Science Publishers, 1986.

[5.12] V. Raghavan, R. A. Rohrer, L. T. Pillage, J. Y. Lee, J. E. Bracken, and M. M. Alaybeyi, "AWE-inspired," in *Proc. IEEE Custom Integrated Circuits Conf.,* May 1993.

[5.13] E. Chiprout and M. S. Nakhla, "Analysis of interconnect networks using complex frequency hopping (CFH)," *IEEE Transactions on Computer-Aided Design,* vol. 14, no.2, pp. 186-200, Feb. 1995.

[5.14] M. Celik and A. C. Cangellaris, "Efficient transient simulation of lossy packaging interconnects using moment-matching techniques," *IEEE Trans. Comp., Packag., Manufact. Technol. PartB: Advanced Packaging,* vol. 19, no. 1, pp. 64-73, Feb. 1996.

[5.15] J. E. Bracken, V. Raghavan, and R. A. Rohrer, "Interconnect simulation using asymptotic waveform evaluation (AWE)," *IEEE Transactions on Circuits and Systems,* vol. 39, no. 11, pp. 869-878, Nov. 1992.

CHAPTER 6

Passive Reduced-Order Multiport Models

Circuit level simulation of interconnects with their (non)linear drivers and receivers can be runtime costly due primarily to the size of the linear interconnect portion of the model. Considering that the number of interactions between interconnect and drivers/receivers (ports) will be small compared to the number of interconnect elements, it is often appropriate to partition the linear interconnect portions from the drivers and receivers. The interconnect can then be macromodeled using an efficient order reduction technique. These macromodels are then combined with other linear blocks and nonlinear components in a circuit simulation environment.

This chapter focuses on the macromodeling part of the problem. We first present linear multiport *Y*-parameter models, followed by various Krylov subspace based projection methods for their reduced order modeling. Among these methods, PRIMA guarantees stability and passivity in addition to providing superior accuracy.

6.1 Multiport Modeling

Consider a simple interconnect problem which consists of a driver, a load, and an interconnect between them. The entire circuit can be partitioned into linear interconnect and (non)linear driver and receiver blocks as pictured in Figure 6.1. Notice that the only connection between the interconnect block and the outside world is a few ter-

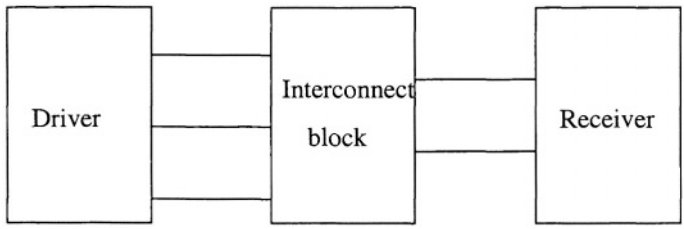


FIGURE 6.1 A typical interconnect problem.

minals. If we are not interested in what is happening inside the block, the only information we need to know about it is the current-voltage characteristics at the terminals. This section concentrates on effective modeling of such interconnect blocks. But first, we give a few definitions.

A *terminal* is a node that is accessible from outside of the block. For each terminal there is also a reference terminal, which provides the return path for the current. A terminal together with its reference is called a *port*. Usually the reference terminal is the ground node. A circuit which is represented with its terminals or ports is called a *multiport*. A general representation of a multiport is shown in Figure 6.2. Such a circuit can be characterized in terms of one of the following interchangeable descriptions: *Y* (admittance), *Z* (impedance), *H* (hybrid), *S* (scattering), or transmission parameters. Since it is more suitable to the MNA formulation, most of the time in this chapter we use *Y*-parameter representations. Definitions of *Y* and Z parameters are given next.

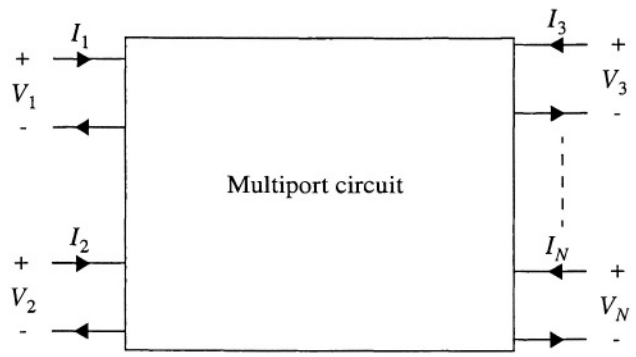


FIGURE 6.2 Circuit with N port.

6.1.1 *Y* and Z parameters

Consider the multiport shown in Figure 6.2. Its port behavior can be described by a matrix equation in the form

$$\mathbf{YV} = \mathbf{I} \tag{6.1}$$

or

$$\begin{bmatrix} Y_{11} & Y_{12} & \cdots & Y_{1N} \\ Y_{21} & Y_{22} & \cdots & Y_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ Y_{N1} & Y_{N2} & \cdots & Y_{NN} \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ V_N \end{bmatrix} = \begin{bmatrix} I_1 \\ I_2 \\ \vdots \\ I_N \end{bmatrix} \tag{6.2}$$

where $\mathbf{Y}$ is the admittance matrix and, V_i and I_i are the port voltages and currents, respectively. To determine Y_{ij}, we apply voltage sources to all ports and short-circuit all of them except the *j*th. We then measure the currents at the ports. Only the *j*th column of the matrix (6.2) remains, yielding

$$Y_{ij}V_j = I_i \qquad i = 1, \ldots, N \tag{6.3}$$

or

$$Y_{ij} = \left.\frac{I_i}{V_j}\right|_{\text{all ports except the jth short-circuited}} \tag{6.4}$$

Similarly, the *Z* parameters are defined as

$$\mathbf{ZI} = \mathbf{V} \tag{6.5}$$

where $\mathbf{Z}$ is the $N \times N$ impedance matrix. To determine Z_{ij}, we apply current sources to all ports and open-circuit all of them except the *j*th. We then measure the voltage across the *i*th port.

The *Y* and *Z* parameters are subsets of the more general so-called *matrix transfer functions*. Given the linear circuit with input/output ports are identified, the matrix transfer functions, and the *Y* and *Z* parameters in particular, can be obtained in a systematic manner using a suitable circuit formulation technique such as the MNA formulation.

6.1.2 Circuit Formulation for Macromodeling

Consider a multi-input multi-output linear circuit and its time-domain MNA description:

$$\begin{aligned} \mathbf{G}\mathbf{x} + \mathbf{C}\frac{d\mathbf{x}}{dt} &= \mathbf{B}\mathbf{u}_{\text{in}}(t) \\ \mathbf{u}_{\text{out}}(t) &= \mathbf{L}^T\mathbf{x} \end{aligned} \tag{6.6}$$

For a formal derivation of the MNA formulation, refer to Section 5.2. Here we provide a brief definition of the terms in (6.6). Let n be the total number of MNA variables, and, N_i and N_o be the number of inputs and outputs, respectively. Then,

$\mathbf{G}$ and $\mathbf{C}$ are the $n \times n$ MNA circuit matrices, representing the conductance and energy storage elements, respectively;

$\mathbf{x}$ is the vector of MNA variables of size n;

$\mathbf{u}_{\text{in}}(t)$ is the vector of input excitations of size N_i;

$\mathbf{B}$ is the $n \times N_i$ source connectivity matrix mapping $\mathbf{u}_{\text{in}}(t)$ to the MNA vector $\mathbf{x}$;

$\mathbf{u}_{\text{out}}(t)$ is the output vector of size N_o;

$\mathbf{L}$ is the $n \times N_o$ probing matrix mapping $\mathbf{u}_{\text{out}}(t)$ to the MNA vector $\mathbf{x}$;

Now let us define the $N_i \times N_o$ matrix transfer function $\mathbf{H}(s)$ as

$$\mathbf{U}_{\text{out}}(s) = \mathbf{H}(s)\mathbf{U}_{\text{in}}(s) \tag{6.7}$$

where $\mathbf{U}_{\text{in}}(s)$ and $\mathbf{U}_{\text{out}}(s)$ are the Laplace transforms of $\mathbf{u}_{\text{in}}(t)$ and $\mathbf{u}_{\text{out}}(t)$, respectively.

From (6.6) and (6.7) it follows that

$$\mathbf{H}(s) = \mathbf{L}^T(\mathbf{G} + s\mathbf{C})^{-1}\mathbf{B} \tag{6.8}$$

Next assume that $\mathbf{G}$ is invertible. Defining

$$\mathbf{A} = -\mathbf{G}^{-1}\mathbf{C} \text{ and } \mathbf{R} = \mathbf{G}^{-1}\mathbf{B} \tag{6.9}$$

we can rewrite the double matrix MNA description in (6.6) in the form of a single matrix representation:

$$\mathbf{x} = \mathbf{A}\frac{d\mathbf{x}}{dt} + \boldsymbol{R}\mathbf{u}_{\text{in}}(t) \qquad \mathbf{u}_{\text{out}}(t) = \mathbf{L}^T\mathbf{x} \tag{6.10}$$

The matrix transfer function $\mathbf{H}(s)$ becomes

$$\mathbf{H}(s) = \mathbf{L}^T(\mathbf{I} - s\mathbf{A})^{-1}\mathbf{R} \tag{6.11}$$

Equation (6.11) can also be written as

$$\mathbf{H}(s) = \frac{\mathbf{L}^T \text{adj}(\mathbf{I} - s\mathbf{A})\mathbf{R}}{\det(\mathbf{I} - s\mathbf{A})} \tag{6.12}$$

where adj and det stand for adjacent and determinant, respectively. From (6.12) it follows that the entries of the matrix $\mathbf{H}(s)$ are in the form of rational polynomials of s :

$$H_{ij}(s) = \frac{b_{ij1} + b_{ij2}s + \ldots + b_{ijm}s^m}{1 + a_2 s + \ldots + a_n s^n} \tag{6.13}$$

The denominator polynomial, $\det(\mathbf{I} - s\mathbf{A})$, is common to all entries, and its roots, reciprocals of the eigenvalues of $\mathbf{A}$, are the poles of the multiport. Equation (6.13) can also be expressed in the pole-zero or pole-residue representation.

In Y parameter formulation, the only sources allowed in the circuit are the voltage sources across the ports. In this case, we replace $\mathbf{u}_{\text{in}}(t)$ with $\mathbf{u}_p(t)$, the vector of port voltages, and $\mathbf{u}_{\text{out}}(t)$ with $\mathbf{i}_p(t)$, the vector of currents flowing into the multiport. Moreover, since the outputs are measured at the inputs, it can be shown that

$$\mathbf{B} = -\mathbf{L} \tag{6.14}$$

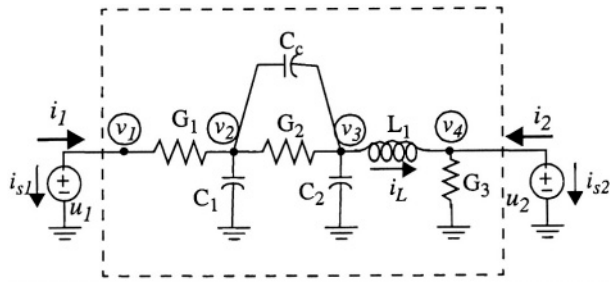
Thus the MNA description becomes

$$\begin{aligned} \mathbf{G}\mathbf{x} + \mathbf{C}\frac{d\mathbf{x}}{dt} &= \mathbf{B}\mathbf{u}_p(t) \\ \mathbf{i}_p(t) &= -\mathbf{B}^T\mathbf{x} \end{aligned} \tag{6.15}$$

and the admittance matrix is given by

$$\mathbf{Y}(s) = -\mathbf{B}^T(\mathbf{G} + s\mathbf{C})^{-1}\mathbf{B} \tag{6.16}$$

An example is given in Figure 6.3. It shows a two-port circuit and the corresponding MNA quantities.



$$\mathbf{G} = \begin{bmatrix} G_1 & -G_1 & 0 & 0 & 0 & 1 & 0 \\ -G_1 & (G_1+G_2) & -G_2 & 0 & 0 & 0 & 0 \\ 0 & -G_2 & G_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & G_3 & -1 & 0 & 1 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \quad \mathbf{C} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & (C_c+C_1) & -C_c & 0 & 0 & 0 & 0 \\ 0 & -C_c & (C_c+C_2) & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -L_1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \mathbf{X} = \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \\ I_L \\ I_{s1} \\ I_{s2} \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\mathbf{U}_p = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \qquad \mathbf{I}_p = \begin{bmatrix} I_1 \\ I_2 \end{bmatrix} \qquad \mathbf{L} = -\mathbf{B}$$

FIGURE 6.3 MNA matrices for Y-parameter formulation of a two port RLC circuit.

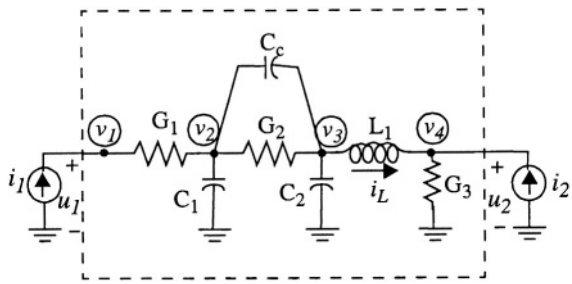
The *Z* parameter formulation is similar:

$$\begin{aligned} \mathbf{G}\mathbf{x} + \mathbf{C}\frac{d\mathbf{x}}{dt} &= \mathbf{B}\mathbf{i}_p(t) \\ \mathbf{u}_p(t) &= \mathbf{B}^T\mathbf{x} \end{aligned} \tag{6.17}$$

Thus

$$\mathbf{Z}(s) = \mathbf{B}^T(\mathbf{G} + s\mathbf{C})^{-1}\mathbf{B} \tag{6.18}$$

The same type of expressions for $\mathbf{Y}(s)$ and $\mathbf{Z}(s)$ in (6.16) and (6.18) may be confusing at first glance; however, even for the same multiport circuit, all three matrices (**G, C, B**) are different. An example of these differences is shown by comparing the *Y* parameter formulation in Figure 6.3 with the *Z* parameter formulation in Figure 6.4 for the same example circuit.



$$\mathbf{G} = \begin{bmatrix} G_1 & -G_1 & 0 & 0 & 0 \\ -G_1 & (G_1+G_2) & -G_2 & 0 & 0 \\ 0 & -G_2 & G_2 & 0 & 0 \\ 0 & 0 & 0 & G_3 & -1 \\ 0 & 0 & 1 & -1 & 0 \end{bmatrix} \quad \mathbf{C} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & (C_c+C_1) & -C_c & 0 & 0 \\ 0 & -C_c & (C_c+C_2) & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -L_1 \end{bmatrix} \quad \mathbf{X} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ I_L \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}$$

$$\mathbf{U}_p = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \quad \mathbf{I}_p = \begin{bmatrix} i_1 \\ i_2 \end{bmatrix} \quad \mathbf{L} = \mathbf{B}$$

FIGURE 6.4 MNA matrices for *Z* parameter formulation of a two-port RLC circuit.

6.1.3 Circuit Formulation Types

We now analyze some properties of the MNA matrices. These properties, such as symmetry and positive definiteness, are important when investigating the accuracy, stability, and passivity characteristics of the multiport model order reduction techniques.

First we define the definiteness of a matrix. A matrix $\mathbf{A}$ is said to be positive definite, denoted by $\mathbf{A} > 0$, if $\mathbf{x}^T\mathbf{A}\mathbf{x} > 0$ for every nonzero vector $\mathbf{x}$. It is positive semidefinite, denoted by $\mathbf{A} \geq 0$, if $\mathbf{x}^T\mathbf{A}\mathbf{x} \geq 0$ for every nonzero vector $\mathbf{x}$.

We categorize linear circuits into the following four types depending on the properties of the corresponding MNA matrices.

A. Symmetric Formulation of RLC Circuits:

The standard MNA formulation for RLC circuits was derived in Chapter 5. It was shown that the MNA matrices can be partitioned as

$$\mathbf{G} = \begin{bmatrix} \mathbf{A}_g G \mathbf{A}_g^T & \mathbf{A}_l & \mathbf{A}_v \\ \mathbf{A}_l^T & 0 & 0 \\ \mathbf{A}_v^T & 0 & 0 \end{bmatrix} \qquad \mathbf{C} = \begin{bmatrix} \mathbf{A}_c C \mathbf{A}_c^T & 0 & 0 \\ 0 & -L & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{6.19}$$

The matrices $\mathbf{A}_g G \mathbf{A}_g^T$, $\mathbf{A}_c C \mathbf{A}_c^T$, and L are square and symmetric. They are also known to be positive semidefinite. Consequently, the $\mathbf{G}$ and $\mathbf{C}$ matrices are clearly symmetric but indefinite.

For this type of formulation we also require $\mathbf{B} = -\mathbf{L}$ or $\mathbf{B} = \mathbf{L}$. Thus, the impedance and admittance formulations of RLC circuits fall into this category.

B. Passive Formulation of RLC Circuits:

Now change the signs of the second and third blocks in (6.19) to obtain

$$\mathbf{G} = \begin{bmatrix} \mathbf{A}_g G \mathbf{A}_g^T & \mathbf{A}_l & \mathbf{A}_v \\ -\mathbf{A}_l^T & 0 & 0 \\ -\mathbf{A}_v^T & 0 & 0 \end{bmatrix} \qquad \mathbf{C} = \begin{bmatrix} \mathbf{A}_c C \mathbf{A}_c^T & 0 & 0 \\ 0 & L & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{6.20}$$

The matrix **G** is not symmetric anymore, but it can easily be shown that both matrices have become positive semidefinite. This transformation is important in both preserving and proving the passivity property in PRIMA. Note that although both (6.19) and (6.20) describe the same passive RLC circuit, it is very difficult, if not impossible, to prove the passivity with (6.19).

Similar to the symmetric formulation, this type of formulation is also restricted to the circuits with $\mathbf{B} = \mathbf{L}$.

C. Symmetric and Passive Formulation of RC and RL Circuits:

In this case we only consider the *Z*-parameter modeling. In the impedance modeling of RC circuits, the **G** and **C** matrices simply become

$$\mathbf{G} = \mathbf{A}_g G \mathbf{A}_g^T \qquad \mathbf{C} = \mathbf{A}_c C \mathbf{A}_c^T \tag{6.21}$$

with $\mathbf{B} = \mathbf{L}$. Thus both **G** and **C** are symmetric and positive semidefinite.

It is possible to obtain similar symmetric and passive formulations for RL and LC circuits.

D. General Linear Circuits

Some of the methods mentioned in this chapter do not pose any restrictions on the matrices (**G**, **C**, **B**, **L**). An obvious example for this type is the general matrix transfer function formulation in (6.10) and (6.11). Clearly, such a flexibility is very advantageous. However, most provably stable and passive methods lose their appealing properties for this most general case.

6.2 Macromodeling Using AWE

Consider again a linear *N*-port. We have shown that the multiport admittance function of such a circuit can be represented as a matrix of transfer functions in the pole-residue form

$$Y_{ij}(s) = \sum_{m=1}^{n} \frac{k_{ijm}}{s - p_m}, \qquad \text{for } 1 \le i, j \le N \tag{6.22}$$

In Chapter 4 we introduced Asymptotic Waveform Evaluation (AWE) as a technique to find Padé approximations of circuit transfer functions. Thus, a macromodel of a linear multiport can be obtained by applying AWE to the entries of its admittance matrix. That is, each entry in (6.16) is modeled by a reduced order AWE approximation:

$$\hat{Y}_{ij}(s) = \sum_{m=1}^{q_{ij}} \frac{k_{ijm}}{s - p_{ijm}}, \qquad \text{for } 1 \le i, j \le N \tag{6.23}$$

It is apparent that once the moments of $Y_{ij}(s)$ are computed, finding the poles and residues is a straightforward application of moment matching. A different order can be used for each approximation. Alternatively, all the entries may be forced to have a common set of poles. In this case, an AWE approximation is determined only for one entry. The same set of poles is then used to find the residues for all the other entries applying partial Padé approximations. For different macromodeling approaches that employ explicit moment matching techniques, refer to [6.1] [6.2] [6.3].

The cost of finding moments for an entire multiport is a single LU factorization and the required number of forward and back substitutions. Consider the *block moments* of $\mathbf{Y}(s)$ which are the coefficients of the Taylor expansion of $\mathbf{Y}(s)$ around $s = 0$:

$$\mathbf{Y}(s) = \mathbf{Y}_0 + \mathbf{Y}_1 s + \mathbf{Y}_2 s^2 + \ldots . \tag{6.24}$$

The ith block moment

$$\mathbf{Y}_i = \mathbf{L}^T \mathbf{A}^i \mathbf{R} \tag{6.25}$$

is an $N \times N$ matrix. It is easy to show that these block moments can be computed recursively using the relation

$$\begin{aligned} \mathbf{G}\mathbf{H}_k &= -\mathbf{C}\mathbf{H}_{k-1} \qquad k > 0 \\ \mathbf{Y}_k &= \mathbf{L}^T \mathbf{H}_k \end{aligned} \tag{6.26}$$

with $\mathbf{GH}_0 = \mathbf{B}$. It follows from (6.26) that we need only one LU factorization for the first moment vector in the first block. All others are computed using simple forward and backward substitutions. The *k*th moment of $Y_{ij}(s)$ is then obtained as

$$m_{kij} = [\mathbf{Y}_k]_{ij} \tag{6.27}$$

As explained in Chapter 4, despite its nice properties, AWE suffers from certain numerical limitations. Recall that these problems are mainly due to moment calculation and explicit moment matching. More precisely, we have shown that the sequence

$$\mathbf{R}, \mathbf{AR}, \ldots, \mathbf{A}^i\mathbf{R}, \ldots, \tag{6.28}$$

which we use to generate moments, converges rapidly to the eigenvector of the largest eigenvalue of the matrix $\mathbf{A}$. Thus, including more moments does not add extra information to the reduced order model.

The remainder of this chapter presents a different approach for the order reduction problem. The methods based on this approach do not use numerically ill-conditioned explicit moment matching. Instead, they employ projection, or congruence transformation, to construct reduced order models. In addition, rather than moments, they use Krylov vectors, which are numerically better conditioned. We introduce the Krylov subspaces in the next section and the projection method in Section 6.4.

6.3 Krylov Subspaces

Before we begin, we point out that to understand Krylov subspaces and projection methods some basic linear algebra knowledge is required. In this chapter, we assume that the reader has a working knowledge of this subject and we review a basic concept only when necessary. Some good references are [6.4] for linear algebra, [6.5] for computational issues, and [6.6] for linear system considerations.

Before introducing the Krylov subspaces, we first provide a few basic definitions from linear algebra.

A subset of a vector space is called a *subspace*. Given a set of vectors $\mathbf{V}_n = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$, the set of all linear combinations of these vectors is a subspace referred to as the *span* of $\mathbf{V}_n$:

$$\text{span}\{\mathbf{V}_n\} = \text{span}\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\} = \left\{ \mathbf{x} \mid \mathbf{x} = \sum_{j=1}^{n} \beta_j \mathbf{v}_j \right\} \tag{6.29}$$

where β_j's are real numbers. If the $\mathbf{v}_j$'s are linearly independent, then each vector of $\text{span}\{\mathbf{V}_n\}$ admits a unique expression as a linear combination of the $\mathbf{v}_j$'s. The set $\mathbf{V}_n$ is called a *basis* of the subspace $\text{span}\{\mathbf{V}_n\}$.

Given an $n \times n$ matrix $\mathbf{A}$ and a vector $\mathbf{r}$, the *Krylov subspace* $\text{Kr}(\mathbf{A}, \mathbf{r}, q)$ is defined as

$$\text{Kr}(\mathbf{A}, \mathbf{r}, q) = \text{span}\{\mathbf{r}, \mathbf{A}\mathbf{r}, \mathbf{A}^2\mathbf{r}, \ldots, \mathbf{A}^{q-1}\mathbf{r}\} \tag{6.30}$$

Consider an $n \times q$ rectangle matrix $\mathbf{V}_q$ whose columns form bases for the subspace spanned by the Krylov sequence $\{\mathbf{r}, \mathbf{A}\mathbf{r}, \mathbf{A}^2\mathbf{r}, \ldots, \mathbf{A}^{q-1}\mathbf{r}\}$, that is,

$$\text{colsp}\mathbf{V}_q = \text{Kr}(\mathbf{A}, \mathbf{r}, q) \tag{6.31}$$

where colsp$\mathbf{M}$ denotes the *column space* of $\mathbf{M}$. Equation (6.31) is equivalent to saying that for each $i = 0, 1, \ldots, q-1$ there exists a q-dimensional column vector β_i such that

$$\mathbf{A}^i\mathbf{r} = \mathbf{V}_q\beta_i. \tag{6.32}$$

Later in this chapter we explain how to generate the basis matrix $\mathbf{V}_q$ via two numerically robust methods: Arnoldi process and Lanczos algorithm. But for now, we provide a simple interpretation of the Krylov vectors, the columns of the matrix $\mathbf{V}_q$, in the circuit analysis context. The details of this subject are discussed at different places throughout the chapter. A brief discussion follows:

Consider a linear circuit with a single excitation and its MNA description of $\{\mathbf{G}, \mathbf{C}, \mathbf{b}\}$. By defining $\mathbf{A} = -\mathbf{G}^{-1}\mathbf{C}$ and $\mathbf{r} = \mathbf{G}^{-1}\mathbf{b}$, it is straightforward to show that the moment vectors of this circuit are given by $\mathbf{A}^i\mathbf{r}$, $i = 0, 1, \ldots$. Assume that a

basis matrix $\mathbf{V}_q$ is generated for the Krylov subspace Kr($\mathbf{A}$, $\mathbf{r}$, q). Equation (6.32) clearly shows how the columns of $\mathbf{V}_q$ are related to the circuit moments. Any moment vector can be expressed as a linear combination of the Krylov vectors. Roughly speaking, these vectors contain the same information. However, the Krylov vectors contain much less numerical noise compared to the circuit moments because during the generation of a Krylov vector the effects of lower-order moment vectors are implicitly subtracted. We will also see that calculating Krylov vectors is as simple and efficient as calculating circuit moments.

The Krylov subspace in (6.30) is defined for a single starting vector. Similarly, given an $n \times n$ matrix $\mathbf{A}$ and an $n \times N$ matrix $\mathbf{R}$, the *block Krylov subspace* is defined as

$$\mathrm{Kr}(\mathbf{A}, \mathbf{R}, q) = \mathrm{span}\{\mathbf{R}, \mathbf{AR}, \ldots, \mathbf{A}^{k-1}\mathbf{R}\} \tag{6.33}$$

where $k = q/N$. If q/N does not result as an integer, we set[1] $k = \lfloor q/N \rfloor$ and define the Krylov subspace as

$$\mathrm{Kr}(\mathbf{A}, \mathbf{R}, q) = \mathrm{span}\{\mathbf{R}, \mathbf{AR}, \ldots, \mathbf{A}^{k-1}\mathbf{R}, \mathbf{A}^{k}\mathbf{r}_1, \mathbf{A}^{k}\mathbf{r}_2, \ldots, \mathbf{A}^{k}\mathbf{r}_l\} \tag{6.34}$$

where $\mathbf{r}_i$ is the ith column vector of $\mathbf{R}$, and $l = q - kN$. For the sake of simplicity, however, we will always assume that q/N is an integer.

Now consider an $n \times q$ rectangle matrix $\mathbf{V}_q$, whose columns form bases for the subspace spanned by the Krylov sequence $\{\mathbf{R}, \mathbf{AR}, \ldots, \mathbf{A}^{k-1}\mathbf{R}\}$, that is,

$$\mathrm{colsp}\,\mathbf{V}_q = \mathrm{Kr}(\mathbf{A}, \mathbf{R}, q). \tag{6.35}$$

Thus, there exist $q \times N$ matrices β_i such that

$$\mathbf{A}^{i}\mathbf{R} = \mathbf{V}_q\beta_i, \qquad i = 0, 1, \ldots, k-1 \tag{6.36}$$

Analogous to the single input case, equation (6.36) shows the relation between the Krylov matrix and the block moments of a circuit with multiple excitations (refer to equations (6.6), (6.9), and (6.28)).

1. The $\lfloor . \rfloor$ operator is the truncation to the nearest integer towards zero.

6.4 Projection Methods for Order Reduction

We now introduce the projection methods. Iterative projection methods have long been used in linear system solutions [6.7] and have recently become popular for model order reduction [6.8] [6.9] [6.10]. Here we will introduce the basic concepts to establish the background for the following sections, where we combine projection methods with Krylov subspaces to obtain robust and accurate order reduction techniques. For a detailed review of projection method, refer to [6.7].

Consider a linear system

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{6.37}$$

where $\mathbf{A}$ is an $n \times n$ real matrix. *Projection* techniques extract an approximate solution to the above system from a search subspace K of dimension q so that q constraints are satisfied. Generally, these constraints are described as imposing q independent orthogonality conditions. For example, the residual vector $\mathbf{Ax} - \mathbf{b}$ is constrained to be orthogonal to q linearly independent vectors. This defines another subspace L of dimension q. Such constraints are known as Petrov-Galerkin conditions.

There are two classes of projection methods: if the subspace K is the same as L, the projection is said to be *orthogonal*; otherwise, it is an *oblique* projection.

In terms of linear dynamic systems, the projection is associated with matrix transformations. For example, consider a single-input single-output linear circuit and its governing equations in terms of $n \times n$ MNA matrices

$$\mathbf{G}\mathbf{x} + \mathbf{C}\frac{d\mathbf{x}}{dt} = \mathbf{b}u_{\text{in}}(t) \qquad u_{\text{out}}(t) = \mathbf{l}^T\mathbf{x} \tag{6.38}$$

Defining $\mathbf{A} = -\mathbf{G}^{-1}\mathbf{C}$ and $\mathbf{r} = \mathbf{G}^{-1}\mathbf{b}$, we obtain

$$\mathbf{x} = \mathbf{A}\frac{d\mathbf{x}}{dt} + \mathbf{r}u_{\text{in}}(t) \qquad u_{\text{out}}(t) = \mathbf{l}^T\mathbf{x} \tag{6.39}$$

Consider two q-dimensional subspaces K and L. Let $\mathbf{V}_q$ be an $n \times q$ matrix whose column vectors form a basis of K, and similarly let $\mathbf{W}_q$ be an $n \times q$ matrix whose column vectors form a basis of L, i.e.

$$\text{colsp}\mathbf{V}_q = K \qquad \text{colsp}\mathbf{W}_q = L \tag{6.40}$$

Then a reduced order model for the system in (6.39) via projection is given as

$$\begin{aligned} \mathbf{W}_q^T\mathbf{V}_q\mathbf{x}_q &= \mathbf{W}_q^T\mathbf{A}\mathbf{V}_q\frac{d\mathbf{x}_q}{dt} + \mathbf{W}_q^T\mathbf{r}u_{\text{in}}(t) \\ u_{\text{out}}(t) &= \mathbf{l}^T\mathbf{V}_q\mathbf{x}_q \end{aligned} \tag{6.41}$$

Since the approximation order, q is smaller than the number of original variables, n, the system in (6.41) is a reduced-order approximation to the original in (6.39), that is, the output response $u_{\text{out}}(t)$ is an approximation to the actual output response $u_{\text{out}}(t)$ in (6.39). In projection terms, the $q \times q$ matrix $\mathbf{W}_q^T\mathbf{A}\mathbf{V}_q$ is the projection of $\mathbf{A}$ onto the subspace spanned by $\mathbf{V}_q$, and orthogonal to the subspace spanned by $\mathbf{W}_q$. Similarly, one can think that the solution vector is approximated by another solution vector, but in the subspace K,

$$\mathbf{x} = \mathbf{V}_q\mathbf{x}_q \tag{6.42}$$

Alternatively, the original system in (6.38) can be reduced with double matrix projection (reducing $\mathbf{G}$ and $\mathbf{C}$ separately):

$$\begin{aligned} \mathbf{W}_q^T\mathbf{G}\mathbf{V}_q\mathbf{x}_q + \mathbf{W}_q{}^T\mathbf{C}\mathbf{V}_q\frac{d\mathbf{x}_q}{dt} &= \mathbf{W}_q^T\mathbf{b}u_{\text{in}}(t) \\ u_{\text{out}}(t) &= \mathbf{l}^T\mathbf{V}_q\mathbf{x}_q \end{aligned} \tag{6.43}$$

In model order reduction via projection, the approximate solution is sought in the subspace $K = \text{span}(\mathbf{V}_q)$ and the residual is orthogonal to the subspace $L = \text{span}(\mathbf{W}_q)$, that is, it satisfies the Petrov-Galerkin conditions. The first condition is apparent from (6.42). For the second condition, consider the residual for (6.38) in the Laplace domain which is given by

$$\mathbf{e} = \mathbf{b} - (\mathbf{G} + s\mathbf{C})\mathbf{x} \tag{6.44}$$

where we have assumed that $U_{\text{in}}(s) = 1$. Multiplying (6.44) by $\mathbf{W}_q^T$ and using (6.42) and (6.43) yields

$$\mathbf{W}_q^T\mathbf{e} = \mathbf{W}_q^T\mathbf{b} - \mathbf{W}_q^T(\mathbf{G} + s\mathbf{C})\mathbf{V}_q\mathbf{x}_q = \mathbf{0}, \tag{6.45}$$

which means that the residual $\mathbf{e}$ is orthogonal to $L = \text{span}(\mathbf{W}_q)$. Therefore, the reduced order system in (6.43) satisfies the Petrov-Galerkin conditions. A similar derivation is also possible for (6.41).

So far we have explained model order reduction via projection for single-input single-output systems, and we have used arbitrary subspaces. Next we introduce projections onto Krylov subspaces. We consider two methods for this purpose: block Lanczos and block Arnoldi. These methods are developed for multi-input multi-output linear dynamic systems so that they can be used for multiport interconnect macromodeling.

Specifically, we take a system in the form

$$\mathbf{G}\mathbf{x} + \mathbf{C}\frac{d\mathbf{x}}{dt} = \mathbf{B}\mathbf{u}_{\text{in}}(t) \qquad \mathbf{u}_{\text{out}}(t) = \mathbf{L}^T\mathbf{x} \tag{6.46}$$

or

$$\mathbf{x} = \mathbf{A}\frac{d\mathbf{x}}{dt} + \mathbf{R}\mathbf{u}_{\text{in}}(t) \qquad \mathbf{u}_{\text{out}}(t) = \mathbf{L}^T\mathbf{x} \tag{6.47}$$

where $\mathbf{A} = -\mathbf{G}^{-1}\mathbf{C}$ and $\mathbf{R} = \mathbf{G}^{-1}\mathbf{B}$, and then approximate it via projection:

$$\begin{aligned} \mathbf{W}_q^T\mathbf{V}_q\mathbf{x}_q &= \mathbf{W}_q^T\mathbf{A}\mathbf{V}_q\frac{d\mathbf{x}_q}{dt} + \mathbf{W}_q^T\mathbf{R}\mathbf{u}_{\text{in}}(t) \\ \mathbf{u}_{\text{out}}(t) &= \mathbf{L}^T\mathbf{V}_q\mathbf{x}_q \end{aligned} \tag{6.48}$$

6.4.1 Block Arnoldi

The first algorithm we describe to produce bases for Krylov subspaces is the block-Arnoldi process. It was originally introduced to reduce a dense matrix to block upper Hessenberg form [6.11]. The block Arnoldi process recursively produces an orthonormal basis for the Krylov subspace generated by a given matrix $\mathbf{A}$ and a starting block of vectors $\mathbf{R}$. The basic algorithm is as follows:

Algorithm (Block Arnoldi):

1. Compute QR factorization $\mathbf{R} = \mathbf{V}_0\mathbf{X}$
2. For $j = 1, 2, \ldots$
3. Compute $\mathbf{V}_j^{(0)} = \mathbf{A}\mathbf{V}_{j-1}$
4. For $i = 1, 2, \ldots, j$
5. Compute $\mathbf{H}_{j-i,j-1} = \mathbf{V}_{j-i}^T\mathbf{V}_j^{(i-1)}$
6. Compute $\mathbf{V}_j^{(i)} = \mathbf{V}_j^{(i-1)} - \mathbf{V}_{j-i}\mathbf{H}_{j-i,j-1}$
7. EndFor
8. Compute QR factorization $\mathbf{V}_j^{(j)} = \mathbf{V}_j\mathbf{H}_{j,j-1}$
9. EndFor
10. Set $\mathbf{V} = \left[\mathbf{V}_0\ \mathbf{V}_1\ \ldots\right]$ and $\mathbf{H} = (\mathbf{H}_{ij})$

The above steps will be explained individually when we present the Arnoldi-based PRIMA algorithm in Subsection 6.7.1. For now, we briefly show some properties of the items generated by the Arnoldi algorithm.

Assume that $\mathbf{A}$ is $n \times n$ and $\mathbf{R}$ is $n \times N$. After k iterations the block Arnoldi process produces an $n \times q$ matrix [1]

$$\mathbf{V}_q = \left[\mathbf{V}_0\ \mathbf{V}_1\ \ldots\ \mathbf{V}_{k-1}\right]. \tag{6.49}$$

whose columns form bases for the Krylov subspace generated by $\mathbf{A}$ and $\mathbf{R}$. Namely,

$$\text{colsp}(\mathbf{V}_q) = \text{Kr}(\mathbf{A}, \mathbf{R}, q) \tag{6.50}$$

Furthermore, its columns can be shown to be orthonormal

$$\mathbf{V}_q^T\mathbf{V}_q = \mathbf{I} \tag{6.51}$$

The algorithm also generates a $q \times q$ block upper Hessenberg matrix

1. The subscript q is reserved for denoting a basis matrix with q columns. Any other subscript i denotes the ith block in $\mathbf{V}_q$.

$$\mathbf{H}_q = \begin{bmatrix} \mathbf{H}_{00} & \mathbf{H}_{01} & \cdots & \mathbf{H}_{0,k-1} \\ \mathbf{H}_{10} & \mathbf{H}_{11} & \cdots & \mathbf{H}_{1,k-1} \\ \vdots & \ddots & \ddots & \vdots \\ \mathbf{0} & \cdots & \mathbf{H}_{k-1,k-2} & \mathbf{H}_{k-1,k-1} \end{bmatrix} \tag{6.52}$$

which satisfies

$$\mathbf{V}_q^T \mathbf{A} \mathbf{V}_q = \mathbf{H}_q \tag{6.53}$$

and

$$\mathbf{A}\mathbf{V}_q = \mathbf{V}_q \mathbf{H}_q + \mathbf{V}_k \mathbf{H}_{k,k-1} \begin{bmatrix} \mathbf{0} & \mathbf{0} & \cdots & \mathbf{I} \end{bmatrix} \tag{6.54}$$

where $\mathbf{I}$ is a square identity matrix with the appropriate size.

Now we return to the problem of model order reduction via projection. Assume that $\mathbf{V}_q$ is generated for the matrices $\mathbf{A}$ and $\mathbf{R}$ in (6.47). Applying an orthogonal projection, the reduced order model in (6.48) becomes

$$\mathbf{x}_q = \mathbf{H}_q \frac{d\mathbf{x}_q}{dt} + \mathbf{V}_q^T \mathbf{R} \mathbf{u}_{\text{in}}(t) \qquad \mathbf{u}_{\text{out}}(t) = \mathbf{L}^T \mathbf{V}_q \mathbf{x}_q \tag{6.55}$$

Thus the reduced-order admittance matrix is given by

$$\mathbf{Y}_q(s) = \mathbf{L}^T \mathbf{V}_q (\mathbf{I} - s\mathbf{H}_q)^{-1} \mathbf{V}_q^T \mathbf{R}. \tag{6.56}$$

Its moments are

$$\mathbf{Y}_{q,i} = \mathbf{L}^T \mathbf{V}_q \mathbf{H}_q^i \mathbf{V}_q^T \mathbf{R}. \tag{6.57}$$

The moment-matching property of the block Arnoldi for general linear circuits is given by the following theorem. But first we present a lemma whose proof is given in Appendix 6.A:

Lemma 6.1: *Let* $\mathbf{V}_q$ *be an Arnoldi generated basis matrix for* Kr($\mathbf{A}$, $\mathbf{R}$, q), *then*

$$\mathbf{A}^i\mathbf{R} = \mathbf{V}_q\mathbf{H}_q^i\mathbf{V}_q^T\mathbf{R}, \qquad i = 0, 1, \ldots, k-1 \tag{6.58}$$

Theorem 6.2: *The block Arnoldi method preserves the first $k = q/N$ block moments, that is,*

$$\mathbf{L}^T\mathbf{A}^i\mathbf{R} = \mathbf{L}^T\mathbf{V}_q\mathbf{H}_q^i\mathbf{V}_q^T\mathbf{R}, \qquad i = 0, 1, \ldots, k-1 \tag{6.59}$$

Proof: The proof follows from Lemma 6.1. Multiplying both sides with $\mathbf{L}^T$ yields (6.59). QED

A special case for this algorithm is noteworthy. For RLC circuits with symmetric formulation, the matrix $\mathbf{A} = -\mathbf{G}^{-1}\mathbf{C}$ is symmetric, which implies that the reduced order matrix $\mathbf{H}_q = \mathbf{V}_q^T\mathbf{A}\mathbf{V}_q$ will also be symmetric. In other words, $\mathbf{H}_q$ will be block tridiagonal. This causes a simplification in the algorithm: $\mathbf{H}_{j-i,j-1}$ will be zero for $j \geq 3$, so that the orthogonalizations in steps 4-7 need to be executed only for $i = 1, 2$. Furthermore, the moment matching accuracy is also superior in this case. Consider the following theorem:

Theorem 6.3: *For RLC circuits with symmetric formulation (see Subsection 6.1.3), the block Arnoldi method preserves the first $2k$ block moments, that is,*

$$\mathbf{B}^T\mathbf{A}^i\mathbf{R} = \mathbf{B}^T\mathbf{V}_q\mathbf{H}_q^i\mathbf{V}_q^T\mathbf{R}, \qquad i = 0, 1, \ldots, 2k-1 \tag{6.60}$$

Proof: By assumption, $\mathbf{G}$, $\mathbf{C}$, $\mathbf{A} = -\mathbf{G}^{-1}\mathbf{C}$, and $\mathbf{H}_q = \mathbf{V}_q^T\mathbf{A}\mathbf{V}_q$ are all symmetric, and $\mathbf{B} = -\mathbf{L}$. Under these assumptions, Lemma 6.1 can be manipulated to obtain

$$\mathbf{B}^T\mathbf{A}^i = \mathbf{B}^T\mathbf{V}_q\mathbf{H}_q^i\mathbf{V}_q^T, \qquad i = 0, 1, \ldots, k-1 \tag{6.61}$$

Choose any i and j such that $i < k$ and $j < k$. Then we can write the moments of the reduced system as

$$\mathbf{B}^T\mathbf{V}_q\mathbf{H}_q^{i+j}\mathbf{V}^T\mathbf{R} = \mathbf{B}^T\mathbf{V}_q\mathbf{H}_q^i\mathbf{H}_q^j\mathbf{V}_q^T\mathbf{R} = \mathbf{B}^T\mathbf{V}_q\mathbf{H}_q^i\mathbf{V}_q^T\mathbf{V}_q\mathbf{H}_q^j\mathbf{V}_q^T\mathbf{R}. \tag{6.62}$$

From Lemma 6.1 and (6.61) it follows that

$$\mathbf{B}^T\mathbf{V}\mathbf{H}_q^i\mathbf{V}_q^T\mathbf{V}_q\mathbf{H}_q^j\mathbf{V}_q^T\mathbf{R} = \mathbf{B}^T\mathbf{A}^i\mathbf{A}^j\mathbf{R}, \qquad i < k, \quad j < k \text{ QED} \tag{6.63}$$

6.4.2 Block Lanczos

The other technique we will describe to generate bases for Krylov subspaces is the Lanczos algorithm. It was originally proposed by Lanczos [6.12] as a method for the computation of eigenvalues of symmetric and nonsymmetric matrices. The Lanczos algorithm starts with a matrix $\mathbf{A}$ and two blocks of vectors $\mathbf{R}$, $\mathbf{L}$, and proceeds to generate two sequences of block matrices

$$\mathbf{V}_q = \left[\mathbf{V}_0\ \mathbf{V}_1\ \ldots\ \mathbf{V}_{k-1}\right] \qquad \text{and} \qquad \mathbf{W}_q = \left[\mathbf{W}_0\ \mathbf{W}_1\ \ldots\ \mathbf{W}_{k-1}\right] \tag{6.64}$$

which are bases for the Krylov subspaces

$$\text{colsp}(\mathbf{V}_q) = \text{Kr}(\mathbf{A}, \mathbf{R}, q) \qquad \text{and} \qquad \text{colsp}(\mathbf{W}_q) = \text{Kr}(\mathbf{A}^T, \mathbf{L}, q) \tag{6.65}$$

The following algorithm is taken from [6.13].

Algorithm (Block Lanczos):

1. Compute factorizations $\mathbf{V}_0\beta_0 = \mathbf{R}$ and $\mathbf{W}_0\Gamma_0 = \mathbf{L}$
2. For $j = 1, 2, \ldots$ while $\mathbf{V}_j \neq \mathbf{0}$ and/or $\mathbf{W}_j \neq \mathbf{0}$
3. Set $\mathbf{V}_j^{(0)} = \mathbf{A}\mathbf{V}_{j-1}$ and $\mathbf{W}_j^{(0)} = \mathbf{A}^T\mathbf{W}_{j-1}$
4. For $i = 1, 2$
5. Set $\mathbf{T}_{j-i,j-1} = \mathbf{D}_{j-i}^{-1}\mathbf{W}_{j-i}^T\mathbf{V}_j^{(i-1)}$ and $\mathbf{V}_j^{(i)} = \mathbf{V}_j^{(i-1)} - \mathbf{V}_{j-i}\mathbf{T}_{j-i,j-1}$
6. Set $\tilde{\mathbf{T}}_{j-i,j-1} = \mathbf{D}_{j-i}^{-1}\mathbf{V}_{j-i}^T\mathbf{W}_j^{(i-1)}$ and $\mathbf{W}_j^{(i)} = \mathbf{W}_j^{(i-1)} - \mathbf{W}_{j-i}\tilde{\mathbf{T}}_{j-i,j-1}$
7. EndFor
8. Compute factorizations $\mathbf{V}_j\beta_j = \mathbf{V}_j^{(2)}$ and $\mathbf{W}_j\Gamma_j = \mathbf{W}_j^{(2)}$
9. Set $\mathbf{D}_j = \mathbf{W}_j^T\mathbf{V}_j$
10. EndFor

In steps 1 and 8, the normalization matrices β_j and Γ_j are generally chosen to make $\mathbf{D}_j = \mathbf{W}_j^T\mathbf{V}_j = \mathbf{I}$. For example, if we choose, as proposed in [6.9],

$$\Gamma_j^T\beta_j = \text{LU factors of } (\mathbf{W}_j^{(2)})^T\mathbf{V}_j^{(2)} \tag{6.66}$$

it follows that

$$\mathbf{I} = (\Gamma_j^T)^{-1}(\mathbf{W}_j^{(2)})^T\mathbf{V}_j^{(2)}\beta_j^{-1} = \mathbf{W}_j^T\mathbf{V}_j = \mathbf{D}_j. \tag{6.67}$$

However, if $(\mathbf{W}_j^{(2)})^T\mathbf{V}_j^{(2)}$ turns out to be singular, a breakdown error occurs [6.5]. Recently, a robust block Lanczos algorithm was given in [6.14] which includes a built-in deflation procedure to detect and delete linearly dependent vectors in the Krylov sequence, and the option to employ look-ahead to avoid the potential breakdowns.

After k iterations the matrices generated by the block Lanczos algorithm satisfy

$$\begin{aligned} \mathbf{A}\mathbf{V}_q &= \mathbf{V}_q\mathbf{T}_q + \mathbf{V}_k\beta_k\begin{bmatrix}\mathbf{0} & \mathbf{0} & \dots & \mathbf{I}\end{bmatrix} \\ \mathbf{A}^T\mathbf{W}_q &= \mathbf{W}_q\tilde{\mathbf{T}}_q + \mathbf{W}_k\Gamma_k\begin{bmatrix}\mathbf{0} & \mathbf{0} & \dots & \mathbf{I}\end{bmatrix} \\ \mathbf{W}_q^T\mathbf{V}_q &= \mathbf{D}_q \\ \mathbf{W}_q^T\mathbf{A}\mathbf{V}_q &= \mathbf{D}_q\mathbf{T}_q \end{aligned} \tag{6.68}$$

where

$$\mathbf{D}_q = \operatorname{diag}(\mathbf{D}_0, \dots, \mathbf{D}_{k-1}), \tag{6.69}$$

$$\mathbf{T}_q = \begin{bmatrix} \mathbf{T}_{00} & \mathbf{T}_{01} & \dots & \mathbf{0} \\ \beta_1 & \mathbf{T}_{11} & \dots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{T}_{k-2,k-1} \\ \mathbf{0} & \dots & \beta_{k-1} & \mathbf{T}_{k-1,k-1} \end{bmatrix}, \text{ and } \tilde{\mathbf{T}}_q = \begin{bmatrix} \tilde{\mathbf{T}}_{00} & \tilde{\mathbf{T}}_{01} & \dots & \mathbf{0} \\ \Gamma_1 & \tilde{\mathbf{T}}_{11} & \dots & \vdots \\ \vdots & \ddots & \ddots & \tilde{\mathbf{T}}_{k-2,k-1} \\ \mathbf{0} & \dots & \Gamma_{k-1} & \tilde{\mathbf{T}}_{k-1,k-1} \end{bmatrix} \tag{6.70}$$

Once the block Lanczos algorithm is run with the matrices $\mathbf{A}$, $\mathbf{R}$, and $\mathbf{L}$ in (6.47), a projection with $\mathbf{V}_q$ and $\mathbf{W}_q$ produces the reduced-order model

$$\mathbf{x}_q = \mathbf{T}_q\frac{d\mathbf{x}_q}{dt} + \begin{bmatrix}\beta_0 \\ \mathbf{0}\end{bmatrix}\mathbf{u}_{\text{in}}(t) \qquad \mathbf{u}_{\text{out}}(t) = \begin{bmatrix}\Gamma_0^T & \mathbf{0}\end{bmatrix}\mathbf{D}_q\mathbf{x}_q \tag{6.71}$$

The corresponding reduced-order admittance matrix becomes

$$\mathbf{Y}_q(s) = \begin{bmatrix} \Gamma_0^T & \mathbf{0} \end{bmatrix} \mathbf{D}_q (\mathbf{I} - s\mathbf{T}_q)^{-1} \begin{bmatrix} \beta_0 \\ \mathbf{0} \end{bmatrix} \tag{6.72}$$

From the above equation, the block moments of the reducer order system can be written as

$$\mathbf{Y}_{q,i} = \begin{bmatrix} \Gamma_0^T & \mathbf{0} \end{bmatrix} \mathbf{D}_q \mathbf{T}_q^i \begin{bmatrix} \beta_0 \\ \mathbf{0} \end{bmatrix} \tag{6.73}$$

In [6.15], it is shown that

$$\mathbf{Y}_{q,i} = \mathbf{Y}_i, \qquad i = 0, 1, \ldots, 2k-1 \tag{6.74}$$

where $\mathbf{Y}_i = \mathbf{L}^T \mathbf{A}^i \mathbf{R}$ are the block moments of the original circuit. Hence we have the following theorem:

Theorem 6.4: *The block Lanczos method preserves the first* $2k$ *block moments.*

In other words, the first $2q/N$ block moments are matched for a qth order system, hence $\mathbf{Y}_q(s)$ is a matrix Padé approximation of $\mathbf{Y}(s)$.

The first method that used a block Lanczos algorithm for interconnect macromodeling was MPVL [6.15]. Subsequently, a more computationally efficient version of MPVL, SyMPVL [6.16], was tailored for RLC circuits with symmetric formulation. Later in this chapter, we show that SyMPVL preserves stability and passivity for RC, RL, and LC circuits.

6.5 Stability and Passivity

The Krylov-based projection methods are, in general, well conditioned; however, the reduced-order model of a stable RLC circuit can still have unstable poles. Although it is possible to obtain an asymptotically stable model by simply discarding the unstable poles, passivity, in general, cannot be guaranteed.

A passive system denotes a system that is incapable of generating energy, and hence one that can only absorb energy from the sources used to excite it [6.17]. Passivity is an important property because stable, but not passive macromodels can produce unstable systems when connected to other stable, even passive, loads. A property in classical circuit theory states this fact: *Interconnections ofstable systems may not necessarily be stable; but passive circuits are stable; and arbitrary interconnections of passive circuits are passive, and, therefore, stable* [6.18].

To see that passivity is a practical problem, consider the simple interconnect circuit with a load and a nonlinear driver as shown in Figure 6.5. The interconnect portion is modeled as a two port and its admittance parameters are approximated by fifth order stable Padé approximations. Consider the SPICE simulation results for the combined circuit. The waveform in the subgraph clearly shows the growing oscillations at the output. Also shown in the same figure is the waveform obtained using the PRIMA method, which is the subject of the next section. As seen from the figure, the PRIMA result is stable, and indistinguishable from the exact waveform.

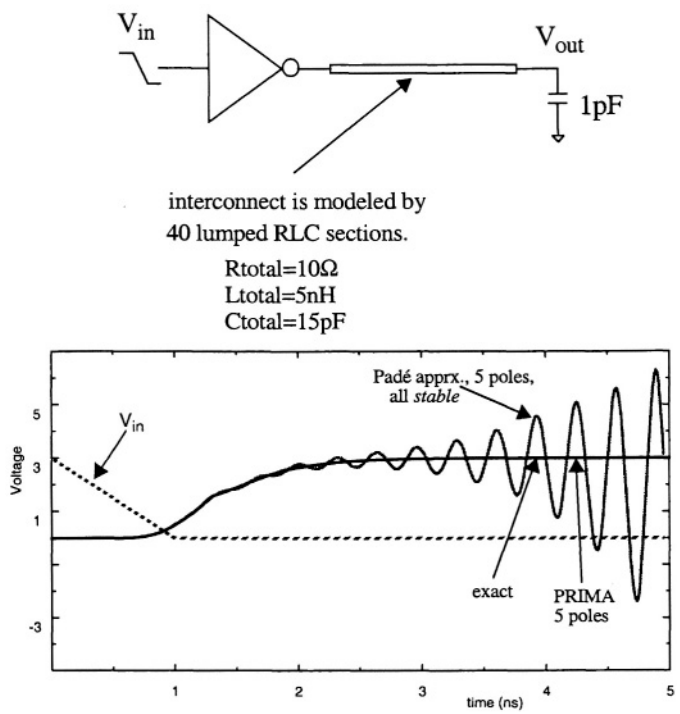


FIGURE 6.5 A stable but not passive macromodel.

PRIMA was proposed by the authors of this book in 1997 [6.19] [6.20] [6.21] as a provably passive reduced order macromodeling method for general RLC interconnect circuits. Approximately at the same time, another provably passive order reduction method was developed. The details of this method, which is based on a concept called Split Congruence Transformations, can be found in [6.22] and [6.23].

We next present the PRIMA method.

6.6 PRIMA

PRIMA (**P**assive **R**educed-order **I**nterconnect **M**acromodeling **A**lgorithm) is a Krylov subspace based projection method which generates guaranteed stable and passive reduced order models. The version we describe here is based on the passive MNA formulation that we explained in Subsection 6.1.3. It is possible, however, to extend it to other type formulations as well, as we discuss in Subsection 6.6.2. In this and the following sections, several properties of the algorithm are investigated and a practical implementation is developed.

Briefly, PRIMA is an orthogonal projection method which takes a linear circuit in the form

$$\mathbf{G}\mathbf{x} + \mathbf{C}\frac{d\mathbf{x}}{dt} = \mathbf{B}\mathbf{u}_p(t) \qquad \mathbf{i}_p(t) = \mathbf{L}^T\mathbf{x} \tag{6.75}$$

and finds a reduced-order model:

$$\mathbf{V}_q^T\mathbf{G}\mathbf{V}_q\mathbf{x}_q + \mathbf{V}_q^T\mathbf{C}\mathbf{V}_q\frac{d\mathbf{x}_q}{dt} = \mathbf{V}_q^T\mathbf{B}\mathbf{u}_p(t) \qquad \mathbf{i}_p(t) = \mathbf{L}^T\mathbf{V}_q\mathbf{x}_q \tag{6.76}$$

Note the difference between PRIMA and the other block Krylov methods: PRIMA employs a double matrix projection whereas the order reduction methods based on Arnoldi and Lanczos use a single matrix projection.

We first describe the simplest implementation of PRIMA which consists of three stages: a specific type of circuit formulation, finding an appropriate Krylov subspace, and projection.

6.6.1 Basic Implementation

A. Circuit Formulation:

In order to preserve passivity, PRIMA requires a minor modification in the MNA formulation of the original circuit. Essentially, PRIMA uses the passive formulation mentioned in Subsection 6.1.3. The details follow:

Consider the *s*-domain MNA formulation of an RLC circuit

$$\begin{bmatrix} \mathbf{A}_g G \mathbf{A}_g^T + s\mathbf{A}_c C \mathbf{A}_c^T & \mathbf{A}_l & \mathbf{A}_v \\ \mathbf{A}_l^T & -sL & \mathbf{0} \\ \mathbf{A}_v^T & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{V}_n \\ \mathbf{I}_l \\ \mathbf{I}_v \end{bmatrix} = \begin{bmatrix} -\mathbf{A}_i \mathbf{I}_i \\ \mathbf{0} \\ \mathbf{V}_v \end{bmatrix} \tag{6.77}$$

For the details of the MNA formulation and the definition of the terms in (6.77), refer to Subsection 5.2.1.

Assume that we are interested in the admittance formulation, therefore the only sources allowed in the circuit are the voltage sources connected to the ports. Consequently, the vector $-\mathbf{A}_i\mathbf{I}_i$ in the right hand side becomes empty. Furthermore, we can express the vector $\mathbf{V}_v$ as $-\mathbf{B}\mathbf{U}_p$, where $\mathbf{U}_p$ is the vector of port voltage sources.

Multiplying both sides of the second and third block of equations in (6.77) by minus one and defining

$$\mathbf{I}_x = \begin{bmatrix} \mathbf{I}_l \\ \mathbf{I}_v \end{bmatrix} \qquad \mathbf{X} = \begin{bmatrix} \mathbf{V}_n \\ \mathbf{I}_x \end{bmatrix} \qquad \mathbf{G}_{12} = \begin{bmatrix} \mathbf{A}_l & \mathbf{A}_v \end{bmatrix}, \tag{6.78}$$

we obtain

$$(\mathbf{G} + s\mathbf{C})\mathbf{X} = \mathbf{B}\mathbf{U}_p \tag{6.79}$$

With this change, the MNA matrices in (6.79) are formed as

$$\mathbf{G} = \begin{bmatrix} \mathbf{G}_{11} & \mathbf{G}_{12} \\ -\mathbf{G}_{12}^T & \mathbf{0} \end{bmatrix} \qquad \mathbf{C} = \begin{bmatrix} \mathbf{C}_{11} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_{22} \end{bmatrix} \tag{6.80}$$

where $\mathbf{G}_{11}$, $\mathbf{C}_{11}$, and $\mathbf{C}_{22}$ are the matrices containing the stamps for resistors, capacitors, and inductors, respectively. In terms of the items given in (6.77), they are

$$\mathbf{G}_{11} = \mathbf{A}_g G \mathbf{A}_g^T \qquad \mathbf{C}_{11} = \mathbf{A}_c C \mathbf{A}_c^T \qquad \mathbf{C}_{22} = \begin{bmatrix} L & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{6.81}$$

From these relations it is apparent that $\mathbf{G}_{11}$, $\mathbf{C}_{11}$, and $\mathbf{C}_{22}$ are symmetric positive semidefinite matrices. Consequently, it follows from (6.80) that $\mathbf{G}$ and $\mathbf{C}$ are also positive semidefinite. In addition, $\mathbf{C}$ is also symmetric.

The reason for this modification will be apparent in Subsection 6.6.3, where we address the passivity issue. An example is given in Figure 6.6, which shows the modified MNA quantities for the example circuit given in Figure 6.3.

B. Finding the Projection Matrix, $\mathbf{V}_q$:

Having obtained the required circuit formulation, the next step is to find the Krylov subspace to be used in the projection. This stage is important for the accuracy and size of the reduced-order system. It is possible to employ different methods to obtain the transformation matrix, $\mathbf{V}_q$, since, as we will see later, the inherent passivity property of PRIMA is independent of the selection of $\mathbf{V}_q$. The simplest way is to use a block Arnoldi algorithm with an expansion around $s = 0$, as proposed originally in [6.20]. But it is also possible to obtain the transformation matrix from an Arnoldi process with multipoint expansion points [6.24] or from a block Lanczos algorithm [6.25]. Unless otherwise stated, for the rest of this chapter it is assumed that the block Arnoldi process is used to generate the transformation matrix $\mathbf{V}_q$.

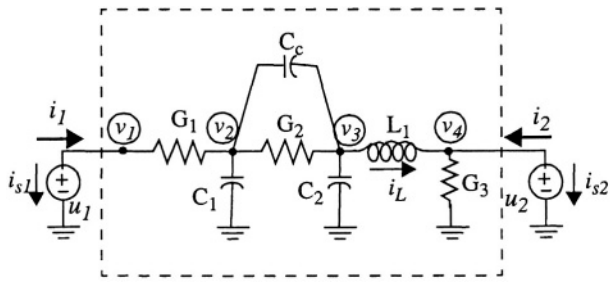
C. The Congruence Transformations to Obtain the Reduced-Order Model:

Once the transformation matrix is obtained, the reduced-order model is constructed as

$$\mathbf{G}_q\mathbf{x}_q + \mathbf{C}_q\frac{d\mathbf{x}_q}{dt} = \mathbf{B}_q\mathbf{u}_p(t)$$
$$\mathbf{i}_p(t) = \mathbf{L}_q^T\mathbf{x}_q \tag{6.82}$$

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

$$\mathbf{C}_q = \mathbf{V}_q^T\mathbf{C}\mathbf{V}_q \qquad \mathbf{G}_q = \mathbf{V}_q^T\mathbf{G}\mathbf{V}_q$$
$$\mathbf{B}_q = \mathbf{V}_q^T\mathbf{B} \qquad \mathbf{L}_q = \mathbf{V}_q^T\mathbf{L} \tag{6.83}$$



$$\mathbf{G} = \begin{bmatrix} G_1 & -G_1 & 0 & 0 & 0 & 1 & 0 \\ -G_1 & (G_1+G_2) & -G_2 & 0 & 0 & 0 & 0 \\ 0 & -G_2 & G_2 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & G_3 & -1 & 0 & 1 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 \end{bmatrix} \quad \mathbf{C} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & (C_c+C_1) & -C_c & 0 & 0 & 0 & 0 \\ 0 & -C_c & (C_c+C_2) & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & L_1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \mathbf{X} = \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \\ I_L \\ I_{s1} \\ I_{s2} \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -1 & 0 \\ 0 & -1 \end{bmatrix}$$

$$\mathbf{U}_p = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \qquad \mathbf{I}_p = \begin{bmatrix} I_1 \\ I_2 \end{bmatrix} \qquad \mathbf{L} = \mathbf{B}$$

FIGURE 6.6 The modified MNA matrices of the example circuit of Figure 6.3

6.6.2 Beyond RLC Circuits

In this book we present PRIMA as an RLC circuit reduction method. However, PRIMA has been successfully applied to some other interconnect problems as well [6.26] [6.27] [6.28]. The crucial point in any application is to find a system description with the symmetry and positive definiteness properties explained in Subsection 6.6.1. Once such a model representation is obtained, applying PRIMA is no different than RLC circuit reduction.

In order to simulate high speed systems, designers need to analyze accurate electromagnetic models of the interconnect and package together with their drivers and receivers in a circuit simulation environment. For many portions of a system, the interconnect may be sufficiently long and uniform so that it can be modeled using a 2-D approximation and the transmission line theory. The *s*-domain models of transmission lines, however, are not compatible with the PRIMA formulation, hence they cannot be used directly. The brute force approach is to model them with multiple RLC segments. But in addition to the possible accuracy loss, this approach also increases the size of the system to be reduced. A more compact and accurate PRIMA compatible transmission line model [6.29] is proposed in [6.26], where a compact finite differences scheme is employed to discretize the transmission line equations. The same approach is also extended to dispersive lines [6.30].

Although a significant portion of the interconnect and package can be modeled with 2-D approximations, discontinuities in the 2-D interconnect, such as vias, bends, and chip to board connectors, require full 3-D modeling. One well known approach to generate circuit models for the 3-D structures is the PEEC method [6.31]. PEEC models are derived from the discretization of the integral formulation for Maxwell's equations, and they can be converted to RLC circuits with some post processing. However, the number of densely coupled circuit elements can be easily in the tens of thousands for a typical 3-D structure for which skin and proximity effects are important. Such PEEC generated circuits are much too expensive to include in a SPICE-like simulator. Thus, PEEC modeling is an area where PRIMA can be very useful by generating reduced-order passive models.

A recent PEEC-based method uses mesh analysis instead of a nodal analysis approach [6.27]. In this method, PRIMA is directly applied to a passive formulation of the full quasistatic Maxwell's equations. The expensive matrix factorizations required to find the Krylov vectors are avoided by employing an iterative method. It is claimed that mesh formulation has better convergence properties than nodal formulation [6.27].

6.6.3 Preservation of Passivity

To prove that PRIMA produces passive and stable reduced order models, we will make use of many theorems and definitions from the classical network synthesis theory. All of the theorems used here, except the ones with proofs, are taken from [6.32].

The first theorem we use relates the passivity of a linear circuit to the positive realness of its associated transfer function:

Theorem 6.5: *A linear dynamic system is passive if and only if the associated transfer function,* $\mathbf{H}(s)$, *is positive-real.*

Thus, all we have to do is to show that the governing matrix transfer function of the system, in our case the admittance matrix $\mathbf{Y}_q(s)$ of the reduced order model, is positive real. The following theorem provides a test for the positive realness:

Theorem 6.6 (Direct positive-realness test): *A matrix function* $\mathbf{H}(s)$ *is positive real if and only if all of the following conditions are satisfied:*

1. $\mathbf{H}(s)$ *is real-rational.*
2. $\mathbf{H}(s)$ *has no poles in* $\mathrm{Re}\{s\} > 0$.
3. *Poles of* $\mathbf{H}(s)$ *on* $\mathrm{Re}\{s\} = 0$ *are simple.*
4. *For each pole on* $\mathrm{Re}\{s\} = 0$, *the residue matrix* $\mathbf{K}$ *is positive semidefinite.*
5. $\mathbf{H}_H(j\omega) = \mathbf{H}(j\omega) + \mathbf{H}^T(-j\omega) \geq \mathbf{0}$ *whenever it is defined*[1].

In any order reduction method we can satisfy the first four conditions (either during the construction of the model or by postprocessing) and we can easily test them. It is, however, very difficult the test the fifth condition. Thus, we need other means to prove the passivity. Consider the following definition of a positive-real matrix function:

Definition 6.7: *A matrix function* $\mathbf{H}(s)$ *is called positive-real if*

1. *Each element of* $\mathbf{H}(s)$ *is analytic for* $\mathrm{Re}\{s\} > 0$.

1. $\mathbf{A}_H(s)$ denotes the Hermitian part of a matrix, $\mathbf{A}(s)$, defined as $\mathbf{A}_H = \mathbf{A}(s) + \mathbf{A}^T(s^*)$

2. $\mathbf{H}(s^*) = \mathbf{H}^*(s)$ *for* $\mathrm{Re}\{s\}>0$.

3. $\mathbf{H}_H(s) = \mathbf{H}(s) + \mathbf{H}^T(s^*) \geq \mathbf{0}$ *for* $\mathrm{Re}\{s\}>0$.

Also consider the following theorems:

Theorem 6.8: *If a matrix,* $\mathbf{A}$, *is positive-real, then so is its inverse, if it exists.*

Theorem 6.9: *If* $\mathbf{A}$ *is positive-real and if* $\mathbf{x}_0^{*T}\mathbf{A}_H\mathbf{x}_0 = 0$ *for some fixed* $\mathbf{x}_0$ *and some fixed* s_0 *with* $\mathrm{Re}\{s_0\} > 0$, *then* $\mathbf{x}_0^{*T}\mathbf{A}_H\mathbf{x}_0 = 0$ *for all* s *with* $\mathrm{Re}\{s\} >0$.

Theorem 6.10: *If* $\mathbf{F}$ *is a real constant* $m \times n$ *matrix and* $\mathbf{A}$ *is an* $m \times m$ *positive-real matrix, then the matrix* $\mathbf{F}^T\mathbf{A}\mathbf{F}$ *is a positive real matrix.*

Now we can state and prove our passivity theorem:

Theorem 6.11: *Let* $\mathbf{G}$ *and* $\mathbf{C}$ *be* $n \times n$, *and* $\mathbf{B}$ $n \times N$ *real matrices. Assume that* $\mathbf{G} + \mathbf{G}^T \geq \mathbf{0}$ *and* $\mathbf{C} = \mathbf{C}^T \geq \mathbf{0}$. *Also assume that* $(\mathbf{G} + s\mathbf{C})$ *is invertible at least at one point* s *with* $\mathrm{Re}\{s\} > 0$. *Then, the matrix transfer function*

$$\mathbf{H}(s) = \mathbf{B}^T(\mathbf{G} + s\mathbf{C})^{-1}\mathbf{B} \tag{6.84}$$

is positive real.

Proof: Let $\mathbf{W}(s) = \mathbf{G} + s\mathbf{C}$. Since the matrices $\mathbf{G}$ and $\mathbf{C}$ are real, the requirements 1) and 2) in Definition 6.7 are automatically satisfied. For the requirement 3), set $s = \sigma + j\omega$ such that $\sigma > 0$. Since $\mathbf{C} = \mathbf{C}^T$, the Hermitian part of $\mathbf{W}$ is

$$\mathbf{W}_h(s) = \mathbf{G} + \mathbf{G}^T + 2\sigma\mathbf{C}. \tag{6.85}$$

Since both $\mathbf{G} + \mathbf{G}^T$ and $\mathbf{C}$ are nonnegative definite, it follows that $\mathbf{W}_h$ is positive semidefinite for $\sigma > 0$, satisfying the third requirement in Definition 6.7. Hence, $\mathbf{W}(s) = \mathbf{G} + s\mathbf{C}$ is positive real.

We next prove that its inverse, $(\mathbf{G}+s\mathbf{C})^{-1}$, exists for $\mathrm{Re}\{s\} > 0$. If we assume that $(\mathbf{G}+s\mathbf{C})^{-1}$ does not exist at s_0 with $\mathrm{Re}\{s_0\} > 0$, then there is an $\mathbf{x}_0$ such that $\mathbf{W}(s_0)\mathbf{x}_0 = 0$, and hence $\mathbf{x}^{*T}_0\mathbf{W}_H(s_0)\mathbf{x}_0 = 0$. But by Theorem 6.9 this Hermitian form is zero for all s with $\mathrm{Re}\{s\} > 0$. This requires $\mathrm{W}(s)$ be singular. But, this contradicts the assumption that $\mathbf{G} + s\mathbf{C}$ is invertible at least for one point s with $\mathrm{Re}\{s\} > 0$. Therefore, $(\mathbf{G}+s\mathbf{C})^{-1}$ exists for $\mathrm{Re}\{s\}>0$.

Finally, in view of Theorems 6.8 and 6.10, $\mathbf{H}(s) = \mathbf{B}^T(\mathbf{G}+s\mathbf{C})^{-1}\mathbf{B}$ is proved to be positive real. QED

With this theorem it is easy to show that the original circuit in (6.79) is passive because the MNA matrices satisfy the assumptions of Theorem 6.11. For the passivity of reduced order models consider the following result:

Corollary 6.12: *Let* $\mathbf{G}$ *and* $\mathbf{C}$ *be* $n \times n$, $\mathbf{B}$ $n \times N$, *and* $\mathbf{V}_q$ $n \times q$ *real matrices. Assume that* $\mathbf{G} + \mathbf{G}^T \geq \mathbf{0}$, $\mathbf{C} = \mathbf{C}^T \geq \mathbf{0}$, *and* $\mathbf{V}_q$ *is full rank. Define* $\mathbf{G}_q = \mathbf{V}_q^T\mathbf{G}\mathbf{V}_q$, $\mathbf{C}_q = \mathbf{V}_q^T\mathbf{C}\mathbf{V}_q$, *and* $\mathbf{B}_q = \mathbf{V}_q^T\mathbf{B}$. *Then, the matrix transfer function*

$$\mathbf{Y}_q(s) = \mathbf{B}_q^T(\mathbf{G}_q + s\mathbf{C}_q)^{-1}\mathbf{B}_q \tag{6.86}$$

is positive real, and thus the reduced order model given by (6.86) is passive.

Proof: From the assumption we have $\mathbf{G} + \mathbf{G}^T \geq \mathbf{0}$, $\mathbf{C} = \mathbf{C}^T \geq \mathbf{0}$, and $\mathbf{V}_q$ is full rank. Thus, $(\mathbf{V}_q^T\mathbf{G}\mathbf{V}_q + \mathbf{V}_q^T\mathbf{G}^T\mathbf{V}_q) \geq \mathbf{0}$ and $\mathbf{V}_q^T\mathbf{C}\mathbf{V}_q = \mathbf{V}_q^T\mathbf{C}^T\mathbf{V}_q \geq \mathbf{0}$. Then, from Theorem 6.11 it follows that (6.86) is positive real. QED

It is important to note that in the proof the only requirement is that the transformation matrix $\mathbf{V}_q$ have full column rank. This gives the algorithm significant flexibility in choosing $\mathbf{V}_q$ to improve macromodel accuracy and runtime.

6.6.4 Preservation of Moments

The moment-matching property of PRIMA is stated by Theorem 6.14. In the proof of the theorem we use of the following lemma, whose proof is given in Appendix 6.A.

Lemma 6.13: *Let* $\mathbf{V}_q$ *be a basis matrix for the Krylov subspace* Kr($\mathbf{A}$, $\mathbf{R}$) *and N be the number of columns in* $\mathbf{R}$. *Then, the matrix*

$$\mathbf{K}_q = -\mathbf{V}_q(\mathbf{V}_q^T\mathbf{G}\mathbf{V}_q)^{-1}\mathbf{V}_q^T\mathbf{C} \tag{6.87}$$

satisfies the relation

$$\mathbf{K}_q^i\mathbf{R} = \mathbf{A}^i\mathbf{R}, \qquad 0 \le i < q/N. \tag{6.88}$$

Theorem 6.14: *Let* $\mathbf{V}_q$ *be a basis matrix for the Krylov subspace* Kr($\mathbf{A}$, $\mathbf{R}$). *Then, the transformation in (6.82) preserves the first k = q/N block moments of the original system.*

Proof: First we prove the relation

$$\mathbf{V}_q(\mathbf{V}_q^T\mathbf{G}\mathbf{V}_q)^{-1}\mathbf{V}_q^T\mathbf{B} = \mathbf{R}. \tag{6.89}$$

Since $\mathbf{V}_q$ is a basis matrix for the Krylov subspace Kr($\mathbf{A}$, $\mathbf{R}$), there exists a matrix α_0 such that

$$\mathbf{R} = \mathbf{V}_q\alpha_0. \tag{6.90}$$

Multiplying both sides of (6.90) with $\mathbf{V}_q(\mathbf{V}_q^T\mathbf{G}\mathbf{V}_q)^{-1}\mathbf{V}_q^T\mathbf{G}$ and using again (6.90), we obtain (6.89).

The moments of the original circuit and the reduced-order models are given by

$$\mathbf{Y}_i = \mathbf{L}^T\mathbf{A}^i\mathbf{R} \qquad \mathbf{Y}_{q,i} = \mathbf{L}_q^T\mathbf{A}_q{}^i\mathbf{R}_q \tag{6.91}$$

where $\mathbf{A}_q = -\mathbf{G}_q^{-1}\mathbf{C}_q$, $\mathbf{R}_q = \mathbf{G}_q^{-1}\mathbf{B}_q$, and $\mathbf{G}_q$, $\mathbf{C}_q$, $\mathbf{R}_q$, and $\mathbf{L}_q$ are as defined in (6.83). From the definitions in (6.83) and (6.87), it follows that

$$\mathbf{Y}_{q,i} = \mathbf{L}^T\mathbf{K}_q^i\mathbf{V}_q(\mathbf{V}_q^T\mathbf{G}\mathbf{V}_q)^{-1}\mathbf{V}_q^T\mathbf{B} \tag{6.92}$$

Substituting (6.89) into (6.92), we obtain

$$\mathbf{Y}_{q,i} = \mathbf{L}^T\mathbf{K}^i\mathbf{R}, \qquad 0 \le i < q/N, \tag{6.93}$$

Finally, from Lemma 6.13 and (6.93) it follows that

$$\mathbf{Y}_{q,i} = \mathbf{L}^T\mathbf{K}^i\mathbf{R} = \mathbf{L}^T\mathbf{A}^i\mathbf{R} = \mathbf{Y}_i, \qquad 0 \le i < q/N, \tag{6.94}$$

which concludes the proof. QED

6.6.5 Accuracy of Double Matrix Projection

In Figure 6.7 a two-bit bus driven by CMOS inverters is shown. One of the drivers is switching while the other is quiet to demonstrate the coupled noise effect. The interconnect, consisting of 40 coupled RLC sections, is modeled as a 4-port and reduced by PRIMA. Transient analysis is done using recursive convolution. The time domain waveforms at the load end are compared for various orders of approximations. Since this is a 4-port, an 8-pole approximation corresponds to matching only m_0 and m_1 generated by four different sources. The plot shows that in the time domain, even the coupled noise can be accurately simulated using the 8 poles from PRIMA.

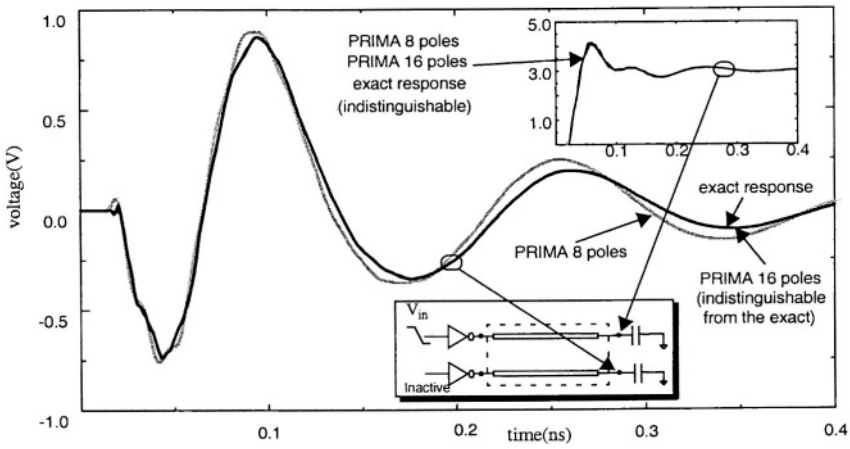


FIGURE 6.7 Waveform comparisons for a four port.

The second example is a 12-port containing six coupled transmission lines modeled by 40 coupled RLC sections. The input admittance $(Y_{11}(s))$, reduced by block Arnoldi, MPVL and PRIMA are compared with the exact input admittance in Figure 6.8 using 48 poles. Block Arnoldi captures the exact response up to 16 GHz, while MPVL and PRIMA match up to 28 GHz. When the order of approximation is increased to 72 poles, it is observed that the frequency spectrum is captured up to 60 GHz by both MPVL and PRIMA.

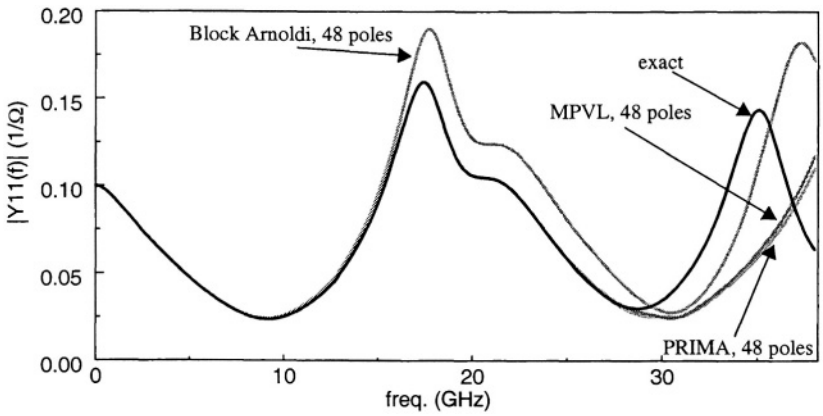


FIGURE 6.8 $Y_{11}(s)$ in frequency domain for six coupled TR. lines.

The above example reveals a surprising accuracy property of PRIMA. As expected, MPVL is more accurate than the block Arnoldi method since it matches twice the number of moments for the same order. But, interestingly, PRIMA's accuracy is also superior to that of the block Arnoldi although it theoretically matches only the same number of moments. This difference in accuracy is attributable to the accuracy loss that appears in the single matrix projection.

Assume that the left and right eigenvectors of **A** are known exactly, that is,

$$\mathbf{A}\mathbf{S}_r = \mathbf{S}_r\Lambda \qquad \mathbf{S}_l^T\mathbf{A} = \Lambda\mathbf{S}_l^T \tag{6.95}$$

where Λ is the diagonal eigenvalue matrix and, $\mathbf{S}_r$ and $\mathbf{S}_l$ are the right and left eigenvector matrices, respectively, with the relation

$$\mathbf{S}_l^T = \mathbf{S}_r^{-1} \tag{6.96}$$

We also make use of

$$\mathbf{C}\mathbf{S}_r = -\mathbf{G}\mathbf{S}_r\Lambda \tag{6.97}$$

which follows from (6.95) and the fact that $\mathbf{A} = -\mathbf{G}^{-1}\mathbf{C}$.

The transformation matrices $\mathbf{V}_q$ and $\mathbf{W}_q$ are approximations to the left and right eigenspaces. If we use the perfect transformation matrices ($\mathbf{V}_q = \mathbf{S}_r$ and $\mathbf{W}_q = \mathbf{S}_l$) to perform the projection, these are the eigenvalues of the reduced system in different reduction methods:

Lanczos:

$$\begin{aligned} \text{eig}(\mathbf{S}_l^T\mathbf{A}\mathbf{S}_r) &= \text{eig}(\mathbf{S}_l^T\mathbf{S}_r\Lambda) \\ &= \text{diag}(\Lambda) \end{aligned} \tag{6.98}$$

Arnoldi:

$$\begin{aligned} \text{eig}(\mathbf{S}_r^T\mathbf{A}\mathbf{S}_r) &= \text{eig}(\mathbf{S}_r^T\mathbf{S}_r\Lambda) \\ &\neq \text{diag}(\Lambda) \end{aligned} \tag{6.99}$$

PRIMA:

$$\begin{aligned} \text{eig}(-(\mathbf{S}_r^T\mathbf{G}\mathbf{S}_r)^{-1}(\mathbf{S}_r^T\mathbf{C}\mathbf{S}_r)) &= \text{eig}((\mathbf{S}_r^T\mathbf{G}\mathbf{S}_r)^{-1}(\mathbf{S}_r^T\mathbf{G}\mathbf{S}_r\Lambda)) \\ &= \text{diag}(\Lambda) \end{aligned} \tag{6.100}$$

Notice the oblique projection in the block Lanczos and orthogonal projections in the block Arnoldi and PRIMA. It can be observed from the derivations that both the Lanczos and PRIMA recover the exact eigenvalues whereas Arnoldi cannot, although both Arnoldi and PRIMA have used the same transformation matrix, $\mathbf{S}_r$. Therefore, it is possible to capture more information by using PRIMA's double matrix projection.

6.6.6 Connections Between Other Projection Methods

PRIMA reduces the two circuit matrices, **G** and **C**, separately, however there are algebraic connections to the single matrix reduction schemes like the Arnoldi and Lanczos algorithms. These connections can be exploited in the implementation of PRIMA for efficiency purposes.

First we define

$$\mathbf{A}_q = -\mathbf{G}_q^{-1}\mathbf{C}_q \qquad \mathbf{R}_q = \mathbf{G}_q^{-1}\mathbf{B}_q, \tag{6.101}$$

and rewrite the PRIMA reduced order models in the form

$$\mathbf{x}_q = \mathbf{A}_q\frac{d\mathbf{x}_q}{dt} + \mathbf{R}_q\mathbf{u}_p \qquad \mathbf{i}_p = \mathbf{L}_q^T\mathbf{x}_q \tag{6.102}$$

A. Relation to the Block Arnoldi Process

Assume that the reduced order matrices in (6.101) and (6.102) are obtained using a block Arnoldi process. Recall the relation (6.54):

$$\mathbf{A}\mathbf{V}_q = \mathbf{V}_q\mathbf{H}_q + \mathbf{V}_{\text{last}} \tag{6.103}$$

where $\mathbf{V}_{\text{last}}$ is defined as

$$\mathbf{V}_{\text{last}} = \mathbf{V}_k\mathbf{H}_{k,k-1}\begin{bmatrix}\mathbf{0} & \mathbf{0} & \dots & \mathbf{I}\end{bmatrix} \tag{6.104}$$

Note that only the last N columns of $\mathbf{V}_{\text{last}}$ are nonzero. In (6.103), $\mathbf{H}_q$ is a $q \times q$ block upper Hessenberg matrix which is the system matrix in the Arnoldi generated reduced order model as shown in (6.55).

Multiplying both sides in (6.103) with $\mathbf{V}_q^T\mathbf{G}$ and since $\mathbf{A} = -\mathbf{G}^{-1}\mathbf{C}$, we obtain

$$-\mathbf{V}_q^T\mathbf{C}\mathbf{V}_q = \mathbf{V}_q^T\mathbf{G}\mathbf{V}_q\mathbf{H}_q + \mathbf{V}_q^T\mathbf{G}\mathbf{V}_{\text{last}} \tag{6.105}$$

or

$$-\mathbf{C}_q = \mathbf{G}_q\mathbf{H}_q + \mathbf{V}_q^T\mathbf{G}\mathbf{V}_{\text{last}} \tag{6.106}$$

Multiplying both sides by $\mathbf{G}_q^{-1}$ reveals the connection between PRIMA and the Arnoldi:

$$\mathbf{A}_q = -\mathbf{G}_q^{-1}\mathbf{C}_q = \mathbf{H}_q + \mathbf{G}_q^{-1}\mathbf{V}_q^T\mathbf{G}\mathbf{V}_{\text{last}} \tag{6.107}$$

Only the last N columns of the $q \times q$ matrix $\mathbf{G}_q^{-1}\mathbf{V}_q^T\mathbf{G}\mathbf{V}_{\text{last}}$ are nonzero because of (6.104). The interpretation of (6.107) is that Arnoldi based PRIMA system matrix is equivalent to the Arnoldi system matrix except the last N columns (see Figure 6.9a).

B. Relation to the Block Lanczos Process

Assume that the transformation matrix $\mathbf{V}_q$ is obtained from the block Lanczos algorithm. Then $\mathbf{V}_q$ satisfies (from (6.68))

$$\mathbf{A}\mathbf{V}_q = \mathbf{V}_q\mathbf{T}_q + \mathbf{V}_{\text{last}} \tag{6.108}$$

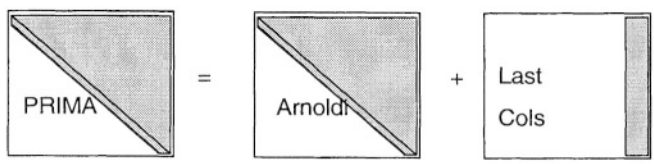
where $\mathbf{V}_{\text{last}}$ is defined as

$$\mathbf{V}_{last} = \mathbf{V}_k\beta_k\begin{bmatrix}\mathbf{0} & \mathbf{0} & \dots & \mathbf{I}\end{bmatrix} \tag{6.109}$$

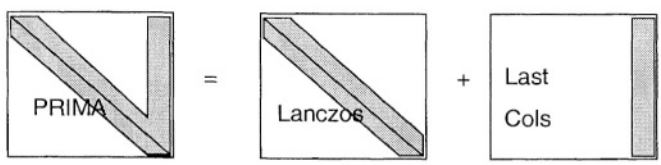
Following a similar algebra as in the Arnoldi case, we obtain the relation between the Lanczos and PRIMA system matrices:

$$\mathbf{A}_q = -\mathbf{G}_q^{-1}\mathbf{C}_q = \mathbf{T}_q + \mathbf{G}_q^{-1}\mathbf{V}_q^T\mathbf{G}\mathbf{V}_{\text{last}}. \tag{6.110}$$

Since $\mathbf{T}_q$ is a tridiagonal matrix, $-\mathbf{G}_q^{-1}\mathbf{C}_q$ is a tridiagonal matrix with a modification for the last N columns as pictured in Figure 6.9 (b).



(a) PRIMA and Arnoldi



(b) PRIMA and Lanczos

FIGURE 6.9 Relations between PRIMA and other Krylov subspace methods.

6.7 Practical Issues

In the previous section we showed that PRIMA takes a linear circuit in the form

$$\mathbf{G}\mathbf{x} + \mathbf{C}\frac{d\mathbf{x}}{dt} = \mathbf{B}\mathbf{u}_p(t) \qquad \mathbf{i}_p(t) = \mathbf{B}^T\mathbf{x} \tag{6.111}$$

and finds a reduced order model:

$$\mathbf{V}_q^T\mathbf{G}\mathbf{V}_q\mathbf{x}_q + \mathbf{V}_q^T\mathbf{C}\mathbf{V}_q\frac{d\mathbf{x}_q}{dt} = \mathbf{V}_q^T\mathbf{B}\mathbf{u}_p(t) \qquad \mathbf{i}_p(t) = \mathbf{B}^T\mathbf{V}_q\mathbf{x}_q \tag{6.112}$$

Furthermore, we proved that the reduced order model in (6.112) is passive if the projection matrix is full rank, i.e.,

$$\text{rank}\,\mathbf{V}_q = q \tag{6.113}$$

We also showed that the reduced model in (6.112) matches the first q/N block moments of the original system if for each $i = 0, 1, \ldots, q/N - 1$ there exists a matrix α_i such that

$$(-\mathbf{G}^{-1}\mathbf{C})^i\mathbf{G}^{-1}\mathbf{B} = \mathbf{V}_q\alpha_i \tag{6.114}$$

The remaining task is to present a robust method to generate a projection matrix $\mathbf{V}_q$ which satisfies (6.113) and (6.114). As we have mentioned earlier, a simple and robust method for this purpose is the block Arnoldi algorithm.

6.7.1 The PRIMA Algorithm

Here we describe a simple practical implementation of PRIMA that employs a block-Arnoldi process with a single point expansion.

Algorithm (Block Arnoldi Based PRIMA): *Given an RLC circuit with the MNA formulation in (6.111), where* **G**, **C**, *and* **B** *are constructed as defined in (6.80) and (6.81), the following algorithm first finds a basis matrix* $\mathbf{V}_q$ *using a block Arnoldi process and then constructs a reduced-order model via projection in the form of (6.112).*

1. Solve for $\mathbf{R}$ in $\mathbf{GR} = \mathbf{B}$
2. Compute QR factorization $\mathbf{R} = \mathbf{V}_0\alpha$ *{block orthonormalization}*
3. For $j = 1, \ldots, k$
4. Solve for $\mathbf{V}_j$ in $\mathbf{GV}_j = -\mathbf{CV}_{j-1}$
5. For $\xi = 1, 2$ *{double orthogonalization}*
6. For $i = 1, \ldots, j$ *{modified Gram-Schmidt orthogonalization}*
7. Set $\Delta = \mathbf{V}_{j-i}^T\mathbf{V}_j$
8. Set $\mathbf{V}_j = \mathbf{V}_j - \mathbf{V}_{j-i}\Delta$
9. Set $\mathbf{H}_{j-i,j-1} = \mathbf{H}_{j-i,j-1} + \Delta$
10. EndFor
11. EndFor
12. Compute QR factorization $\mathbf{V}_j = \mathbf{V}_j\mathbf{H}_{j,j-1}$ *{block orthonormalization}*
13. EndFor
14. Collect generated vector blocks[1] $\mathbf{V}_q = \left[\mathbf{V}_0 \ \ldots \ \mathbf{V}_{k-1}\right]$
15. Set $\mathbf{G}_q = \mathbf{V}_q^T\mathbf{GV}_q$ and $\mathbf{C}_q = \mathbf{V}_q^T\mathbf{CV}_q$

The purpose of steps 1 and 4, which involve operations in terms of the original circuit matrices, is to generate the next block of vectors expanding the Krylov subspace. In Subsection 6.7.3 we give a detailed explanation for efficient evaluations of these steps since they dominate the run time of the algorithm.

Steps 6-10 correspond to a modified Gram-Schmidt orthogonalization of the new block against all the previously generated blocks. Modified Gram-Schmidt is mathematically equivalent to the ordinary Gram-Schmidt process, but has a much better numerical performance [6.5]. The major limitation of the block Arnoldi is the orthogonality loss that occurs between the Krylov vectors as j increases. As step 5 implies, we repeat the Gram-Schmidt orthogonalization to enforce the orthogonality. As we will see a little later, the double orthogonalization is a very effective method to improve the quality of the approximations.

1. We once again emphasize that the subscript q is reserved for denoting a basis matrix with q columns. Any other subscript i will denote the ith block in $\mathbf{V}_q$.

After the new block is orthogonalized against all the previous blocks, steps 2 and 12 use the QR factorization to orthonormalize the vectors in the block. Assuming $r \leq m \leq n$, the QR factorization of an $n \times m$, rank r matrix $\mathbf{M}$ is given by $\mathbf{A} = \mathbf{QR}$, where $\mathbf{Q}$ is an $n \times r$ matrix with orthonormal columns and $\mathbf{R}$ is an $r \times m$ upper triangular matrix. There are several methods to carry out the QR factorization. These methods either are based on a modified Gram-Schmidt orthogonalization process or they apply a sequence of Householder or Givens transformations [6.5].

An important and practical issue in PRIMA is the order selection, which is a common problem in all order reduction methods. There are always two issues associated with the order selection problem. The first one involves finding the minimum order for the desired accuracy. We address this subject in Subsection 6.7.4. The second problem is with the numerical limitation of the algorithm: what is the maximum order that we can obtain from the algorithm? Or perhaps the more important question is, how can we improve the numerical properties of the method?

6.7.2 Improving Numerical Conditioning

Although the Krylov subspace methods are, in general, numerically much more robust than explicit moment matching, they still have limitations on the approximation orders. Due to finite machine precision, the ever-generated Krylov vectors eventually lose orthogonality and the method using them stagnates. The effect of this phenomenon on the accuracy is identical to what happens in AWE: including more Krylov vectors does not necessarily increase the quality of the approximation after some order.

But more importantly, a rank-deficient projection matrix due to orthogonality loss can yield unstable and therefore nonpassive reduced order models. Recall that our passivity proof in Subsection 6.6.3 assumes positive semidefinite reduced order matrices. This assumption in turn requires a full-rank projection matrix. Thus, even if the reduced order model turns out to be stable, we cannot guarantee passivity with a projection matrix that does not possess full rank.

In the case of a rank-deficient projection matrix $\mathbf{V}_q$, the simplest approach is to remove blocks from $\mathbf{V}_q$ starting from the last one until a full rank matrix is obtained. But if we have to use the original set, we can apply a final orthogonalization on $\mathbf{V}_q$ using singular value decomposition.

We next explain two powerful methods to improve the quality of PRIMA approximations:

Frequency shifting: Ironically, the orthogonality loss is related to the convergence of the Krylov vectors to the eigenvectors of the matrix **A**. This is analogous to the dominant pole convergence property of moments. The convergence rate depends on the separateness of the eigenvalues. If they are well separated the Krylov vectors converge quickly to the eigenvectors of **A** starting from the one that corresponds to the largest eigenvalue. Obviously this gives a good approximation around $s = 0$, but the poles near the origin block the effects of the other poles. Thus we can improve the conditioning of Krylov vectors with frequency shifting. With shifting, the poles become less separated and the convergence rate is slowed down. This first deteriorates the quality of the approximation around $s = 0$, but we can generate many more well-conditioned Krylov vectors, and eventually obtain a good approximation in the global sense.

With frequency shifting the PRIMA algorithm is identical to the algorithm in Subsection 6.7.1 except that the steps 1 and 4 are replaced with

$$(\mathbf{G} + s_0\mathbf{C})\mathbf{R} = \mathbf{B} \tag{6.115}$$

and

$$(\mathbf{G} + s_0\mathbf{C})\mathbf{V}_j = \mathbf{C}\mathbf{V}_{j-1}, \tag{6.116}$$

respectively. Usually the expansion point s_0 is a real positive number. For the selection of s_0 and more discussion on frequency shifting, readers are referred to Subsection 4.6.4.

Perhaps the best approach, if we have to apply shifting, is to start generating Krylov vectors at $s = 0$ and then switch to $s = s_0$ when an orthogonality loss is observed. This type of multiple, and even complex, frequency point expansions [6.33] are possible in PRIMA and it extensions [6.24], [6.34], [6.35]. However, we once again emphasize that the improvement with frequency shifting comes with a cost: very often the sparsity of the **G** matrix is destroyed. Therefore, frequency shifting should be avoided if at all possible.

Reorthogonalization: In most cases we can improve the conditioning by simply employing two passes of modified Gram-Schmidt orthogonalization, an approach that has proven to be very effective. The double orthogonalization scheme [6.36], a well-known technique in the Krylov methods, has been observed to produce much higher orders of approximation than the single pass even with expansions at $s = 0$.

We close this subsection by demonstrating the above discussions with an example. The circuit we will consider is a complex two-line interconnect structure. It is modeled as a four port RLC circuit based on the PEEC approach. The size of the MNA matrix is 980 elements and the order of the circuit is 728, i.e., it has 728 finite poles. The **C** matrix is very dense since it includes hundreds of capacitively and inductively coupled elements.

Here we consider only the $Y_{13}(j\omega)$ response. The other responses exhibit similar behavior. In Figure 6.10 the exact response is shown together with the PRIMA result with single orthogonalization and an expansion about $s = 0$. The PRIMA result is obtained from 49 iterations of the block Arnoldi process producing a stable 196th order model. After 49 iterations, PRIMA starts producing unstable poles because of orthogonality loss in the Krylov vectors. Also, as expected following the discussions above, the accuracy does not improve after this 196th order.

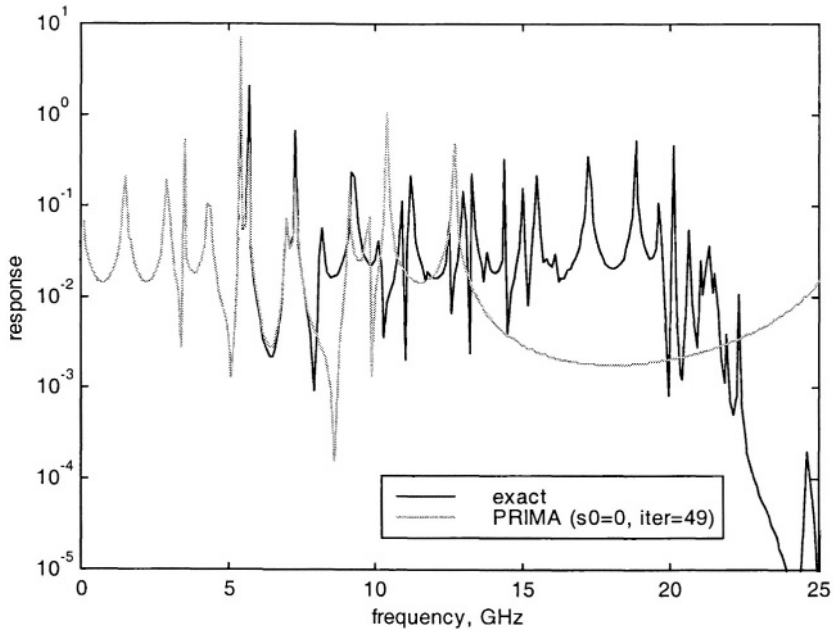


FIGURE 6.10 The comparison of the PRIMA approximation with the exact response. The PRIMA result is obtained with an expansion about $s = 0$ and with single orthogonalization. The order of the PRIMA approximation is 196.

We next consider the PRIMA behavior with double orthogonalization and with an expansion about $s = 0$. With two passes of modified Gram-Schmidt orthogonalization, we do not observe any rank deficiency in the basis matrix until we reach 125 iterations. Thus, we are able to produce a stable 500th order approximation. The response of the reduced model is compared with the exact one in Figure 6.11. It is indistinguishable from the exact response up to 127 GHz.

Consider the PRIMA results with frequency shifting. With single orthogonalization the maximum number of iterations we can get with an expansion about $s_0 = 1.0e11$ is 63, which yields a stable 252th order approximation. With double orthogonalization we can continue until the 80th iteration without seeing any orthogonality loss. The response is shown in Figure 6.11. It matches the exact response very closely up to 95 GHz.

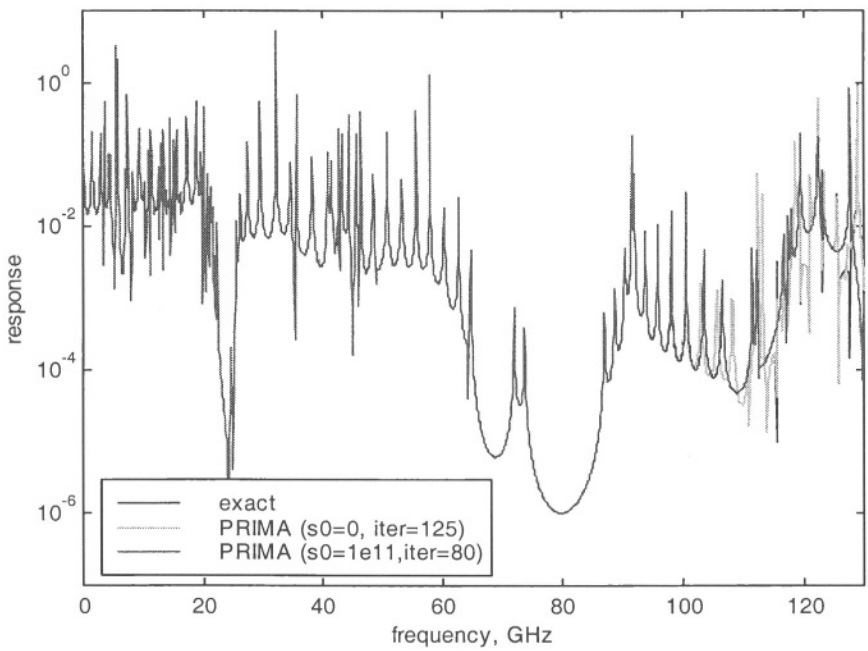


FIGURE 6.11 The comparison of two PRIMA results with the exact response. The first PRIMA approximation (of order 500) is obtained with an expansion about $s = 0$ and with double orthogonalization. The second PRIMA approximation (of order 320) is obtained with an expansion about $s = 1e11$ and with double orthogonalization.

Clearly this is well beyond the frequency range of interest in most IC interconnect problems. But more importantly, what is the runtime cost associated with this level of approximation accuracy?

6.7.3 Calculation of Krylov Vectors

In terms of runtime, the most critical steps in the PRIMA algorithm are the linear system solutions in steps 1 and 4. In general they can be solved with an LU decomposition of the MNA conductance matrix **G**. The Krylov vector generation in linear circuits is similar to moment generation, therefore, all of the techniques we have described in Chapter 5 for moment generation can also be used for efficient Krylov sequence computation.

Recall that the block moments of a linear circuit are obtained recursively using the following algorithm:

$$\begin{aligned} &\mathbf{G}\mathbf{M}_0 = \mathbf{B} \\ &\text{for } j = 1, 2, \ldots, k-1 \\ &\quad \mathbf{G}\mathbf{M}_j = \mathbf{C}\mathbf{M}_{j-1} \\ &\text{end} \end{aligned} \tag{6.117}$$

Thus, for a linear circuit with N ports, in order to generate k block moments, we need to carry out kN forward and back substitutions in addition to a single LU factorization of the usually sparse matrix **G** .

The Krylov vector blocks are obtained from a similar recursive scheme:

$$\begin{aligned} &\mathbf{G}\mathbf{M}_0 = \mathbf{B} \\ &\mathbf{V}_0 = \text{orth}(\mathbf{M}_0) \\ &\text{for } j = 1, 2, \ldots, k-1 \\ &\quad \mathbf{G}\mathbf{M}_j = \mathbf{C}\mathbf{V}_{j-1} \\ &\quad \mathbf{V}_j^{\text{temp}} = \mathbf{M}_j - \mathbf{V}_{j-1}(\mathbf{V}_{j-1}^T\mathbf{M}_j) - \ldots - \mathbf{V}_0(\mathbf{V}_0^T\mathbf{M}_j) \\ &\quad \mathbf{V}_j = \text{orth}(\mathbf{V}_j^{\text{temp}}) \\ &\text{end} \end{aligned} \tag{6.118}$$

which is a condensed and mathematically equivalent version of the PRIMA algorithm given in Subsection 6.7.1. A glance at (6.117) and (6.118) reveals that they require the same type and same number of circuit matrix operations. The PRIMA algorithm has some overhead because of the orthogonalization and orthonormalization procedures. To summarize, in terms of the circuit matrix operations the computational cost of the recursive scheme given in (6.118) is equivalent to that of (6.117).

A. *Implementation with Path Tracing*

The block matrix equation we recursively solve in PRIMA is of the form

$$\mathbf{G}\mathbf{M}_j = \mathbf{C}\mathbf{V}_{j-1} \qquad j = 1, \ldots, k-1 \tag{6.119}$$

Each block in (6.119) requires the solution of N linear systems:

$$\mathbf{G}\mathbf{m}_{j,l} = \mathbf{C}\mathbf{v}_{j-1,l} \qquad j = 1, \ldots, k-1,\ l = 1, \ldots, N \tag{6.120}$$

where $\mathbf{m}_{j,l}$ is the lth column of $\mathbf{M}_j$. As shown in the previous chapter, this can be viewed as the recursive solution of an equivalent dc circuit in which the capacitors and inductors are replaced by current and voltage sources, respectively, with the values derived from the columns of $\mathbf{C}\mathbf{v}_{j-1,l}$. Therefore, the Krylov vectors can be efficiently generated using the path tracing algorithm described in Chapter 5.

Recall, however, that the matrices $\mathbf{G}$ and $\mathbf{C}$ are not explicitly constructed in path tracing. We now describe how we can obtain the matrix products $\mathbf{G}\mathbf{V}_q$ and $\mathbf{C}\mathbf{V}_q$ in this situation. Construction of $\mathbf{C}\mathbf{V}_q$ is straightforward because its columns are the values of the current and voltage sources that are used to replace capacitors and inductors at each moment generation. This information is already available during path tracing. Now consider line 5 in (6.118), which corresponds to the orthogonalization of the new Krylov block against all the previously generated blocks,

$$\mathbf{V}_j^{\text{temp}} = \mathbf{M}_j - \mathbf{V}_{j-1}(\mathbf{V}_{j-1}^T\mathbf{M}_j) - \ldots - \mathbf{V}_0(\mathbf{V}_0^T\mathbf{M}_j) \tag{6.121}$$

The next step in the algorithm, line 6 in (6.118), is the orthonormalization of $\mathbf{V}_j^{\text{temp}}$ using, for example, the QR factorization

$$\mathbf{V}_j^{\text{temp}} = \mathbf{V}_j\mathbf{H}_{j,j-1} \tag{6.122}$$

where $\mathbf{H}_{j,j-1}$ is an $N \times N$ upper triangular nonsingular matrix. Multiplying both sides with $\mathbf{G}$ and substituting (6.121) for $\mathbf{V}_j^{\text{temp}}$, we obtain

$$\mathbf{GV}_j = \mathbf{CV}_{j-1}\mathbf{H}_{j,j-1}^{-1} - \mathbf{GV}_{j-1}(\mathbf{V}_{j-1}^T\mathbf{M}_j)\mathbf{H}_{j,j-1}^{-1} - \ldots - \mathbf{GV}_0(\mathbf{V}_0^T\mathbf{M}_j)\mathbf{H}_{j,j-1}^{-1} \quad (6.123)$$

where we also have substituted (6.119) for $\mathbf{GM}_j$. Thus, the blocks of $\mathbf{GV}_q$ can be computed recursively from the previous blocks of $\mathbf{GV}_q$ and the previous block of $\mathbf{CV}_q$.

The question remaining is, how many times do we apply this recursion; i.e., what is the required order of approximation?

6.7.4 Order Selection and Error Estimation

As we already mentioned, an important step in any order reduction scheme is order selection. Unfortunately, in most methods predicting the order of approximation *a priori* is very difficult. The approach usually applied is to increase the order until a convergence criterion is satisfied.

Ideally we would want to compare the approximated response with the exact one at the desired frequency points. But this is impractical due to the efficiency considerations. A practical and trivial convergence criterion is to check the new reduced order model against the previous lower order approximations. For example, after each block Arnoldi iteration we can construct a reduced order model and obtain the frequency response at some frequency points. For an N-port circuit, the order of the approximation is increased by N with each iteration. We can monitor the changes in the responses and increase the order until successive iterations converge. Obviously, there is no guarantee of always attaining convergence. This uncertainty can be caused by unrealistic frequency point selection. While trying to reach convergence at an arbitrary point we can start generating useless Krylov vectors. Thus any stopping criterion should also monitor the orthogonality loss between the Krylov vectors.

Although the above approach is very practical, having an error estimation of the reduced order model accuracy can still be very useful. In Appendix 6.B we describe an attempt to derive an error estimation that is easy to evaluate.

6.7.5 Post Processing of the Reduced Models

Consider the general representation of a PRIMA reduced model

$$\mathbf{G}_q\mathbf{x} + \mathbf{C}_q\frac{d\mathbf{x}}{dt} = \mathbf{B}_q\mathbf{u}_p \qquad \mathbf{i}_p = \mathbf{L}_q^T\mathbf{x} \tag{6.124}$$

and its admittance matrix

$$\mathbf{Y}_q(s) = \mathbf{L}_q^T(\mathbf{G}_q + s\mathbf{C}_q)^{-1}\mathbf{B}_q . \tag{6.125}$$

In the next chapter we discuss interfacing reduced order models with SPICE-like circuit simulators. In PRIMA, we have two options: 1) we either convert (6.125) into a pole-residue representation and then use a circuit simulator with recursive convolution capability or 2) we transform (6.124) into a state-space representation and then synthesize an equivalent circuit. All these issues including recursive convolution and circuit synthesis are described in detail in the next chapter.

We describe two methods to reduce (6.124) to state-space form

$$\frac{d\mathbf{x}}{dt} = \mathbf{A}_s\mathbf{x} + \mathbf{B}_s\mathbf{u} \qquad \mathbf{i} = \mathbf{C}_s\mathbf{x} \tag{6.126}$$

The first method finds the Cholesky decomposition of $\mathbf{C}_q$ to find a state-space description of the reduced order model. Since it is Cholesky based, this method is computationally very efficient, but usually yields a dense $\mathbf{A}_s$ matrix. The other method is slightly more costly since it is based on eigendecomposition. The benefit is, however, a block diagonal $\mathbf{A}_s$ matrix. This method also gives the poles and the residues of the macromodel

$$\mathbf{Y}_q(s) = \sum_{i=1}^{q}\frac{\mathbf{K}_i}{s - p_i} \tag{6.127}$$

where $\mathbf{K}_i$ is the $N \times N$ residue matrix corresponding to the ith pole of the system

6.8 Special Cases in Multiport Modeling

6.8.1 Symmetric Case (RC, RL, and LC circuits)

For the important special cases of RC, RL, and LC circuits, we can simplify the reduction techniques by exploiting the symmetry seen in the MNA matrices of these types of circuits. In the following, we discuss the RC case. For the case of RL and LC circuits, readers are referred to [6.16].

Consider an RC circuit and its Z parameter formulation:

$$\mathbf{G}\mathbf{x} + \mathbf{C}\frac{d\mathbf{x}}{dt} = \mathbf{B}\mathbf{i}_p(t) \qquad \mathbf{u}_p(t) = \mathbf{B}^T\mathbf{x} \tag{6.128}$$

The impedance matrix is then given by

$$\mathbf{Z}(s) = \mathbf{B}^T(\mathbf{G} + s\mathbf{C})^{-1}\mathbf{B} \tag{6.129}$$

In this case, as we have previously seen in Subsection 6.1.3, the MNA matrices simply become both symmetric and positive semidefinite:

$$\mathbf{G} = \mathbf{A}_g G \mathbf{A}_g^T \qquad \mathbf{C} = \mathbf{A}_c C \mathbf{A}_c^T \tag{6.130}$$

Now assume that $\mathbf{G}$ is positive definite so that its Cholesky factorization exists:

$$\mathbf{G} = \mathbf{G}_L\mathbf{G}_L^T \tag{6.131}$$

where $\mathbf{G}_L$ is lower triangular with positive diagonal entries.

Letting $\mathbf{J} = \mathbf{G}_L^{-1}$ and inserting (6.131) into (6.128) we obtain

$$\mathbf{x} = -\mathbf{J}\mathbf{C}\mathbf{J}^T\frac{d\mathbf{x}}{dt} + \mathbf{J}\mathbf{B}\mathbf{i}_p(t) \qquad \mathbf{u}_p(t) = \mathbf{B}^T\mathbf{J}^T\mathbf{x} \tag{6.132}$$

where the matrix $\mathbf{J}\mathbf{C}\mathbf{J}^T$ is symmetric and positive definite.

Interestingly, for this case, all three methods -- block Arnoldi, block Lanczos, and PRIMA -- yield the same result. Running the block Arnoldi with $\mathbf{A} = -\mathbf{J}\mathbf{C}\mathbf{J}^T$ and $\mathbf{R} = \mathbf{J}\mathbf{B}$ results in a projection matrix $\mathbf{V}_q$ with q columns. Then applying projection to (6.132) results in

$$\mathbf{x}_q = \mathbf{H}_q \frac{d\mathbf{x}_q}{dt} + \mathbf{V}_q^T \mathbf{J} \mathbf{B} \mathbf{i}_p(t) \qquad \mathbf{u}_p(t) = \mathbf{B}^T \mathbf{J}^T \mathbf{V}_q \mathbf{x}_q \tag{6.133}$$

where $\mathbf{H}_q = -\mathbf{V}_q^T \mathbf{J} \mathbf{C} \mathbf{J}^T \mathbf{V}_q$ is a $q \times q$ block tridiagonal and symmetric matrix. In Subsection 6.4.1 we showed that the reduced order system in (6.133) matches the first $2q$ block moments for a qth order system, thereby resulting in a matrix Padé approximation. Hence, the Lanczos and Arnoldi processes produce the same reduced order matrices for the system in (6.132).

A practical problem, however, arises for RC trees with no resistive path to ground. For the impedance modeling of such circuits in which current sources are connected to the ports and voltages at the ports are measured, $\mathbf{G}$ becomes singular. Note that we would have a nonsingular $\mathbf{G}$ matrix, had we used Y parameter formulation. But in this case we could not have obtained the symmetric formulation in (6.132).

The singularity problem in impedance formulation is usually avoided with frequency shifting. In this case, we solve the circuit at $s = s_0$. Unfortunately, frequency shifting, which transforms $\mathbf{G}$ into $\mathbf{G} + s_0\mathbf{C}$, destroys the sparsity of the problem. But we can avoid shifting by employing a practical approach as follows. We can add resistors between the output terminals and the ground node with very large resistance values. Then we can either neglect their effects or compensate for them by adding negative resistances of equal values during the subsequent analyses of the macromodel.

For those who are not satisfied with this practical engineering solution, a more rigorous approach follows. Assume a one-port RC circuit with a singular $\mathbf{G}$ matrix for the sake of explanation. The results can be generalized for the N-port case. Suppose that the nodes are numbered such that the first node in the circuit is the one corresponding to the positive terminal of the port. Thus, the input impedance function of this circuit can be written as

$$Z(s) = \mathbf{e}_1^T (\mathbf{G} + s\mathbf{C})^{-1} \mathbf{e}_1 \tag{6.134}$$

where $\mathbf{e}_1$ is the first unit vector. Now consider adding a parallel resistor between the terminals of the port. It is simple to show that the new conductance matrix is related to matrix $\mathbf{G}$ with a rank one update:

$$\mathbf{G}^{\text{new}} = \mathbf{G} + \frac{1}{R} \mathbf{e}_1 \mathbf{e}_1^T, \tag{6.135}$$

where R is the value of the resistance. Note that $\mathbf{G}^{\text{new}}$ is now invertible because it corresponds to an RC circuit with at least one dc path to ground. Interestingly, in [6.37], it is shown that the span of a Krylov subspace is invariant to certain rank-one

updates such as the one used in (6.135). Thus we can run the Arnoldi process and find a $\mathbf{V}_q$ matrix for the $\{\mathbf{G}^{\text{new}}, \mathbf{C}, \mathbf{e}_1\}$ circuit, but then apply the projection to the $\{\mathbf{G}, \mathbf{C}, \mathbf{e}_1\}$ circuit to find a reduced order model for the impedance function in (6.134).

6.8.2 Single-Input Single-Output Case

In Chapter 4 we introduced AWE -- an explicit moment-matching technique -- to find reduced order models for circuit transfer functions. Likewise, the Krylov subspace based projection techniques can be used to find such approximations as well. In this case, the Krylov vectors are obtained by running the single vector versions of the block Arnoldi and block Lanczos processes given in Section 6.4. We now give a summary of these methods.

We begin with the MNA description of a single-input single-output circuit

$$\mathbf{G}\mathbf{x} + \mathbf{C}\frac{d\mathbf{x}}{dt} = \mathbf{b}u_{\text{in}}(t) \qquad u_{\text{out}}(t) = \mathbf{l}^T\mathbf{x} \tag{6.136}$$

Applying Laplace transformation to (6.136) and letting $s = s' + s_0$, we obtain

$$\begin{aligned}(\mathbf{I} - s'\mathbf{A})\mathbf{X} &= \mathbf{r}U_{\text{in}}(s') \\ U_{\text{out}}(s') &= \mathbf{l}^T\mathbf{X}\end{aligned} \tag{6.137}$$

where $\mathbf{A} = -(\mathbf{G} + s_0\mathbf{C})^{-1}\mathbf{C}$ and $\mathbf{r} = (\mathbf{G} + s_0\mathbf{C})^{-1}\mathbf{b}$. The transfer function of this circuit which we wish to approximate is given by

$$H(s') = \frac{U_{\text{out}}(s')}{U_{\text{in}}(s')} = \mathbf{l}^T(\mathbf{I} - s'\mathbf{A})^{-1}\mathbf{r} \tag{6.138}$$

We first consider the Lanczos process to find an approximation for $H(s')$. The first Krylov subspace based method proposed for circuit analysis was PVL (Padé Via Lanczos) [6.38]. PVL first runs q Lanczos iterations with the matrix $\mathbf{A}$ and the starting vectors $\mathbf{r}$ and $\mathbf{l}$ to obtain a tridiagonal matrix $\mathbf{T}_q$, a diagonal matrix $\mathbf{D}_q$, and two sequences of vectors

$$\mathbf{V}_q = \begin{bmatrix}\mathbf{v}_0 & \mathbf{v}_1 & \cdots & \mathbf{v}_{q-1}\end{bmatrix} \qquad \mathbf{W}_q = \begin{bmatrix}\mathbf{w}_0 & \mathbf{w}_1 & \cdots & \mathbf{w}_{q-1}\end{bmatrix} \tag{6.139}$$

such that

$$\mathbf{W}_q^T \mathbf{A} \mathbf{V}_q = \mathbf{D}_q \mathbf{T}_q \tag{6.140}$$

and

$$\mathbf{W}_q^T \mathbf{V}_q = \mathbf{D}_q = \mathrm{diag}(\delta_0, \delta_1, \ldots, \delta_{q-1}) . \tag{6.141}$$

The first vectors in $\mathbf{V}_q$ and $\mathbf{W}_q$ are selected as

$$\mathbf{v}_0 = \mathbf{r}/\|\mathbf{r}\| \text{ and } \mathbf{w}_0 = \mathbf{l}/\|\mathbf{l}\| \tag{6.142}$$

Applying an oblique projection to the system in (6.137) with the matrices $\mathbf{V}_q$ and $\mathbf{W}_q$ yields a reduced order transfer function of the form

$$H_q(s') = \mathbf{l}^T \mathbf{r} \cdot \mathbf{e}_1^T (\mathbf{I} - s' \mathbf{T}_q)^{-1} \mathbf{e}_1 \tag{6.143}$$

where $\mathbf{e}_1$ is the first unit vector.

From the moment matching property of the block Lanczos method (see Theorem 6.4), it follows that the first $2q$ moments of the transfer functions in (6.138) and (6.143) are matched at $s = s_0$. Therefore, $H_q(s')$ is just the Padé approximation of $H(s')$. This means that with exact arithmetic, hence infinite machine precision, PVL and AWE would produce the same approximations. This is indeed the case for approximation orders up to, typically, $q \approx 6 - 10$. After that AWE begins to suffer from its known numerical problems. PVL, on the other hand, being based on Krylov subspaces, stays numerically stable for much higher orders.

PVL, as a Padé approximation, in general, cannot guarantee stability. However, it has very good convergence properties in the frequency domain. In fact, it was the successful demonstration of PVL's accuracy that has made the applications of Krylov-based techniques popular in linear circuit and electromagnetic analysis [6.39] [6.40] [6.41] [6.42]. We also recognize the work in [6.43], which, approximately at the same time frame as for the PVL work, showed the connection between the Lanczos process and the Padé approximation.

Similar to the Lanczos algorithm, we can also make use of the Arnoldi process to approximate the transfer function in (6.138). In this case, we run the Arnoldi process

with the matrix $\mathbf{A}$ and the starting vector $\mathbf{r}$. After q iterations, we obtain a $q \times q$ upper Hessenberg matrix $\mathbf{H}_q$ and a sequence of vectors

$$\mathbf{V}_q = \left[\mathbf{v}_0\ \mathbf{v}_1\ \cdots\ \mathbf{v}_{q-1}\right] \tag{6.144}$$

such that

$$\mathbf{V}_q^T \mathbf{A} \mathbf{V}_q = \mathbf{H}_q \text{ and } \mathbf{V}_q^T \mathbf{V}_q = \mathbf{I}. \tag{6.145}$$

The first Arnoldi vector is selected as $\mathbf{v}_0 = \mathbf{r}/\|\mathbf{r}\|$. Applying an orthogonal projection to the system in (6.137) with the matrix $\mathbf{V}_q$ yields a reduced order transfer function of the form

$$H_q(s') = \|\mathbf{r}\| \mathbf{l}^T \mathbf{V}_q (\mathbf{I} - s' \mathbf{H}_q)^{-1} \mathbf{e}_1 \tag{6.146}$$

From Theorem 6.2 it follows that the first q moments of (6.138) are matched in $H_q(s')$.

The use of the Arnoldi process in linear circuit analysis was first proposed in [6.39]. Later, it was combined with a modification in the MNA formulation to obtain guaranteed stable transfer function approximations for RLC circuits [6.44], which was the first step towards the passive formulation used in PRIMA.

6.8.3 Single-Input Multi-Output Case

An important difference between the Lanczos- and Arnoldi-based methods is that Lanczos is double-sided. More precisely, a reduced-order model based on a Lanczos algorithm is specifically obtained for a certain input-output pair. On the other hand, the Arnoldi process does not depend on the output. That is, during the generation of Arnoldi vectors we do not use the probing vector or vectors information. Once the matrix $\mathbf{V}_q$ is obtained, we construct transfer function approximations to the desired responses in the original circuit. This property can be exploited when using PRIMA for certain applications.

The following approach can be used in all applications that require stability but not passivity, or only one-port passivity. A typical scenario follows: Consider an interconnect circuit with one driver and $N-1$ receivers. For a SPICE level simulation we need to macromodel the N-port interconnect portion, but now we assume that the receivers can be modeled with linear gate capacitors and only the driver is nonlinear.

This circuit can be efficiently analyzed as follows: First find a driving point model for the interconnect circuit which also includes the fanout capacitances. After the waveform at the driver output is obtained from a circuit simulation of the nonlinear driver loaded by the driving point model, replace the driver with a time-varying voltage source. Then, the output waveforms at the fanouts are found by analyzing the interconnect circuit. This requires only a one-port passive model plus $N-1$ stable transfer functions.

As an example, consider an RLC circuit with a single excitation, and its s-domain MNA formulation

$$(\mathbf{G}+s\mathbf{C})\mathbf{X} = \mathbf{b} \tag{6.147}$$

where the $n \times n$ matrices $\mathbf{G}$ and $\mathbf{C}$ satisfy the required conditions given in Subsection 6.6.1 for a passive reduction. Define a vector of N output responses

$$\mathbf{H}(s) = \begin{bmatrix} H_1(s) \\ H_2(s) \\ \cdots \\ H_N(s) \end{bmatrix} = \mathbf{L}^T\mathbf{X} = \mathbf{L}^T(\mathbf{G}+s\mathbf{C})^{-1}\mathbf{b} \tag{6.148}$$

where $\mathbf{L}$ is the $n \times N$ probing matrix. Let the first column of $\mathbf{L}$ be equal to $\mathbf{b}$ so that

$$H_1(s) = \mathbf{b}^T(\mathbf{G}+s\mathbf{C})^{-1}\mathbf{b} \tag{6.149}$$

is the driving point function.

Also let $\mathbf{V}_q$ be the projection matrix obtained by running the Arnoldi process with $\mathbf{A} = -\mathbf{G}^{-1}\mathbf{C}$ and $\mathbf{r} = \mathbf{G}^{-1}\mathbf{b}$. Now we can construct our reduced-order model:

$$\mathbf{H}_q(s) = \begin{bmatrix} H_{q,1}(s) \\ H_{q,2}(s) \\ \cdots \\ H_{q,N}(s) \end{bmatrix} = \mathbf{L}^T\mathbf{V}_q(\mathbf{V}_q^T\mathbf{G}\mathbf{V}_q + s\mathbf{V}_q^T\mathbf{C}\mathbf{V}_q)^{-1}\mathbf{V}_q^T\mathbf{b} \tag{6.150}$$

where $H_{q,i}(s)$ is a qth order approximation for $H_i(s)$. Clearly, all the transfer function approximations are stable, and in addition, $H_{q,1}(s)$ is a passive driving point model.

The reduced-order model in (6.150) matches the first q moments at each output. Had we used the standard N-port modeling and wanted to match q moments at each port, the order of the reduced model would have been qN. This simple comparison shows the efficiency of using a SIMO (single-input multi-output) version of PRIMA assuming that a fully passive N-port model is not required.

6.9 Summary

In this chapter, we described multiport interconnect modeling using PRIMA. Reduced order models of multiport circuits can be obtained via projections onto Krylov subspaces. Krylov vectors, bases for Krylov subspaces, contain the same information with moments, but numerically better conditioned. Thus their use allow us to obtain very high order accurate reduced order models. In addition, PRIMA exploits the properties of RLC circuit formulation to generate guaranteed stable and passive models.

In the next chapter, we explain how to combine these frequency-domain macromodels with nonlinear drivers and receivers in a SPICE-like simulation environment.

6.A Appendix -- Proofs of the Lemmas

Proof of Lemma 6.1: The proof is by induction. The block Arnoldi process starts with the QR factorization of the starting block of vectors $\mathbf{R}$, that is, an upper triangular matrix α and a matrix $\mathbf{V}_0$ of orthonormal columns are computed such that

$$\mathbf{V}_0 \cdot \alpha = \mathbf{R}. \tag{6.151}$$

Multiplying both sides with $\mathbf{V}_q^T$ and using the orthonormality property of $\mathbf{V}_q$ (see equation (6.51)), we obtain

$$\mathbf{V}_q^T \mathbf{R} = \begin{bmatrix} \alpha \\ \mathbf{0} \\ \cdots \\ \mathbf{0} \end{bmatrix}. \tag{6.152}$$

Multiplying both sides with $\mathbf{V}_q$ and using (6.151) yield

$$\mathbf{R} = \mathbf{V}_q \mathbf{V}_q^T \mathbf{R}, \tag{6.153}$$

which proves (6.58) for $i = 0$.

After k iteration of the block Arnoldi process, we have the relation:

$$\mathbf{A}\mathbf{V}_q = \mathbf{V}_q \mathbf{H}_q + \mathbf{V}_k \mathbf{H}_{k,k-1} \begin{bmatrix} \mathbf{0} & \mathbf{0} & \dots & \mathbf{I} \end{bmatrix} \tag{6.154}$$

Multiplying both sides of (6.154) from right with $\mathbf{H}_q^{i-1}\mathbf{V}_q^T\mathbf{R}$, we obtain

$$\mathbf{A}\mathbf{V}_q\mathbf{H}_q^{i-1}\mathbf{V}_q^T\mathbf{R} = \mathbf{V}_q\mathbf{H}_q^i\mathbf{V}_q^T\mathbf{R} + \mathbf{V}_k\mathbf{H}_{k,k-1}\begin{bmatrix} \mathbf{0} & \mathbf{0} & \dots & \mathbf{I} \end{bmatrix}\mathbf{H}_q^{i-1}\mathbf{V}_q^T\mathbf{R} \tag{6.155}$$

Because of (6.152), the second term on the right hand side is zero. Thus,

$$\mathbf{A}\mathbf{V}_q\mathbf{H}_q^{i-1}\mathbf{V}_q^T\mathbf{R} = \mathbf{V}_q\mathbf{H}_q^i\mathbf{V}_q^T\mathbf{R} \tag{6.156}$$

Now we can finish the proof. Assume that (6.58) is true for $i - 1$, that is,

$$\mathbf{A}^{i-1}\mathbf{R} = \mathbf{V}_q\mathbf{H}_q^{i-1}\mathbf{V}_q^T\mathbf{R} \tag{6.157}$$

Multiplying both sides of (6.157) with $\mathbf{A}$ and then using (6.156) yield (6.58).

Proof of Lemma 6.13: The proof is by induction. It is trivial to prove that (6.88) is true for $i = 0$. Now assume that (6.88) is true for $i - 1$, that is,

$$\mathbf{K}_q^{i-1}\mathbf{R} = \mathbf{A}^{i-1}\mathbf{R}, \qquad 0 \le i < q/N \tag{6.158}$$

Multiplying both sides with $\mathbf{A} = -\mathbf{G}^{-1}\mathbf{C}$ yields

$$-\mathbf{G}^{-1}\mathbf{C}\mathbf{K}_q^{i-1}\mathbf{R} = \mathbf{A}^i\mathbf{R}, \qquad 0 \le i < q/N \tag{6.159}$$

From the assumption of the lemma, $\mathbf{V}_q$ be a basis matrix for the Krylov subspace $\mathrm{Kr}(\mathbf{A}, \mathbf{R})$. Thus for each $i = 0, 1, \dots, q/N - 1$, there exists an $N \times N$ matrix α_i such that

$$\mathbf{A}^i\mathbf{R} = \mathbf{V}_q\alpha_i \tag{6.160}$$

Substituting (6.160) into (6.159) and multiplying both sides in (6.159) with $\mathbf{V}_q(\mathbf{V}_q{}^T\mathbf{G}\mathbf{V}_q)^{-1}\mathbf{V}_q{}^T\mathbf{G}$, we obtain

$$-\mathbf{V}_q(\mathbf{V}_q{}^T\mathbf{G}\mathbf{V}_q)^{-1}\mathbf{V}_q{}^T\mathbf{C}\mathbf{K}_q^{i-1}\mathbf{R} = \mathbf{V}_q\alpha_i \qquad 0 \le i < q/N \tag{6.161}$$

From the definition of $\mathbf{K}_q$ in (6.87) and from (6.160) it follows that

$$\mathbf{K}_q^i\mathbf{R} = \mathbf{A}^i\mathbf{R}, \qquad 0 \le i < q/N, \tag{6.162}$$

which completes the proof of the relation in (6.88).

6.B Appendix -- Error Estimation in PRIMA

In the following, an approximate error measure is derived for PRIMA. However, it is not very suitable to measure the error directly, rather it is useful as a guide to understand the convergence properties.

Recently PVL-WEB [6.45] showed such an error measure for the PVL algorithm, but the region of validity was quite limited. In a similar manner, the exact error of PRIMA transfer function is derived here. This exact error measure is then approximated into an efficient form using a simple heuristic.

We start with rewriting (6.107)

$$\mathbf{A}_q = \mathbf{H}_q + \mathbf{G}_q^{-1}\mathbf{V}_q^T\mathbf{G}\mathbf{V}_{\text{last}}, \tag{6.163}$$

which shows the relation between the system matrices of block Arnoldi and PRIMA. Multiplying both sides with $\mathbf{V}_q$ and rearranging terms yield

$$\mathbf{V}_q\mathbf{H}_q = \mathbf{V}_q\mathbf{A}_q - \mathbf{V}_q\mathbf{G}_q^{-1}\mathbf{V}_q^T\mathbf{G}\mathbf{V}_{\text{last}} \tag{6.164}$$

From (6.103), $\mathbf{V}_q\mathbf{H}_q = \mathbf{A}\mathbf{V}_q - \mathbf{V}_{\text{last}}$. Substituting this into (6.164) and multiplying both sides with s, we obtain

$$s\mathbf{A}\mathbf{V}_q = s\mathbf{V}_q\mathbf{A}_q - s\mathbf{V}_q\mathbf{G}_q^{-1}\mathbf{V}_q^T\mathbf{G}\mathbf{V}_{\text{last}} + s\mathbf{V}_{\text{last}} \tag{6.165}$$

Subtracting $\mathbf{V}_q$ from each side in (6.165) yields

$$(\mathbf{I}-s\mathbf{A})\mathbf{V}_q = \mathbf{V}_q(\mathbf{I}-s\mathbf{A}_q) + s(\mathbf{V}_q\mathbf{G}_q^{-1}\mathbf{V}_q^T\mathbf{G}\mathbf{V}_{\text{last}} - \mathbf{V}_{\text{last}}) \tag{6.166}$$

Multiplying both sides with $(\mathbf{I}-s\mathbf{A})^{-1}$ and $(\mathbf{I}-s\mathbf{A}_q)^{-1}$, from left and right, respectively,

$$\mathbf{V}_q(\mathbf{I}-s\mathbf{A}_q)^{-1} = (\mathbf{I}-s\mathbf{A})^{-1}\mathbf{V}_q + s(\mathbf{I}-s\mathbf{A})^{-1}(\mathbf{V}_q\mathbf{G}_q^{-1}\mathbf{V}_q^T\mathbf{G}\mathbf{V}_{\text{last}} - \mathbf{V}_{\text{last}})(\mathbf{I}-s\mathbf{A}_q)^{-1}$$

This time multiplying both sides with $\mathbf{V}_q^T\mathbf{R}$ and $\mathbf{L}^T$, from right and left, respectively,

$$\begin{aligned}\mathbf{L}^T\mathbf{V}_q(\mathbf{I}-s\mathbf{A}_q)^{-1}\mathbf{V}_q^T\mathbf{R} = {} & \mathbf{L}^T(\mathbf{I}-s\mathbf{A})^{-1}\mathbf{V}_q\mathbf{V}_q^T\mathbf{R} + \\ & s\mathbf{L}^T(\mathbf{I}-s\mathbf{A})^{-1}(\mathbf{V}_q\mathbf{G}_q^{-1}\mathbf{V}_q^T\mathbf{G}\mathbf{V}_{\text{last}} - \mathbf{V}_{\text{last}})(\mathbf{I}-s\mathbf{A}_q)^{-1}\mathbf{V}_q^T\mathbf{R}\end{aligned} \tag{6.167}$$

Since $\mathbf{V}_q\mathbf{V}_q^T\mathbf{R} = \mathbf{R}$ from Lemma 6.1 and recognizing

$$\mathbf{Y}_q(s) = \mathbf{L}^T\mathbf{V}_q(\mathbf{I}-s\mathbf{A}_q)^{-1}\mathbf{V}_q^T\mathbf{R} \text{ and } \mathbf{Y}(s) = \mathbf{L}^T(\mathbf{I}-s\mathbf{A})^{-1}\mathbf{R}, \tag{6.168}$$

the exact error of PRIMA matrix transfer function is given by

$$\begin{aligned}\mathbf{E}(s) &= \mathbf{Y}_q(s) - \mathbf{Y}(s) \\ &= s\mathbf{L}^T(\mathbf{I}-s\mathbf{A})^{-1}(\mathbf{V}_q\mathbf{G}_q^{-1}\mathbf{V}_q^T\mathbf{G}\mathbf{V}_{\text{last}} - \mathbf{V}_{\text{last}})(\mathbf{I}-s\mathbf{A}_q)^{-1}\mathbf{V}_q^T\mathbf{R}\end{aligned} \tag{6.169}$$

Evaluation of (6.169) is not practical because of the $(\mathbf{I}-s\mathbf{A})^{-1}$ term. To compute the error bound for PVL, [6.45] replaces $(\mathbf{I}-s\mathbf{A})^{-1}$ by $1/(1-|s|\|\mathbf{A}\|)$ using of the fact that

$$\left|(\mathbf{I}-s\mathbf{A})^{-1}\right| \le \frac{1}{1-|s|\|\mathbf{A}\|} \text{ for } |s| < 1/\|\mathbf{A}\|. \tag{6.170}$$

A similar approximation could be applied for (6.169) as well; however, the condition in (6.170) dictates a very narrow region for typical high frequency circuit applications. Preconditioning $\mathbf{A}$ to enlarge the validity of (6.170) is possible [6.45] but causes an increase in the overall cost of reduction process. Instead, we choose to replace $(\mathbf{I}-s\mathbf{A})^{-1}$ in (6.169) with $\mathbf{V}_q(\mathbf{I}-s\mathbf{A}_q)^{-1}\mathbf{V}_q^T$. Although this is not a bound, it is very useful in determining the convergence behavior. Therefore, the approximate error measure becomes

$$\mathbf{E}_q(s) = s\mathbf{L}^T\mathbf{V}_q(\mathbf{I}-s\mathbf{A}_q)^{-1}\mathbf{V}_q^T(\mathbf{V}_q\mathbf{G}_q^{-1}\mathbf{V}_q^T\mathbf{G}\mathbf{V}_{\text{last}} - \mathbf{V}_{\text{last}})(\mathbf{I}-s\mathbf{A}_q)^{-1}\mathbf{V}_q^T\mathbf{R} \tag{6.171}$$

Since $\mathbf{V}_q^T\mathbf{V}_{\text{last}} = \mathbf{0}$ and $\mathbf{V}_q^T\mathbf{V}_q = \mathbf{I}$, due to the orthonormality property of the Krylov vectors, we obtain

$$\mathbf{E}_q(s) = s\mathbf{L}^T\mathbf{V}_q(\mathbf{I} - s\mathbf{A}_q)^{-1}\mathbf{G}_q^{-1}\mathbf{V}_q^T\mathbf{G}\mathbf{V}_{\text{last}}(\mathbf{I} - s\mathbf{A}_q)^{-1}\mathbf{V}_q^T\mathbf{R} \tag{6.172}$$

At a specific order of approximation, equation (6.172) can be used to estimate the region of convergence. For example consider the circuit used in Subsection 6.7.2. In Figure 6.12, the approximate error measure (6.172) is plotted for two different PRIMA approximations for the same response.

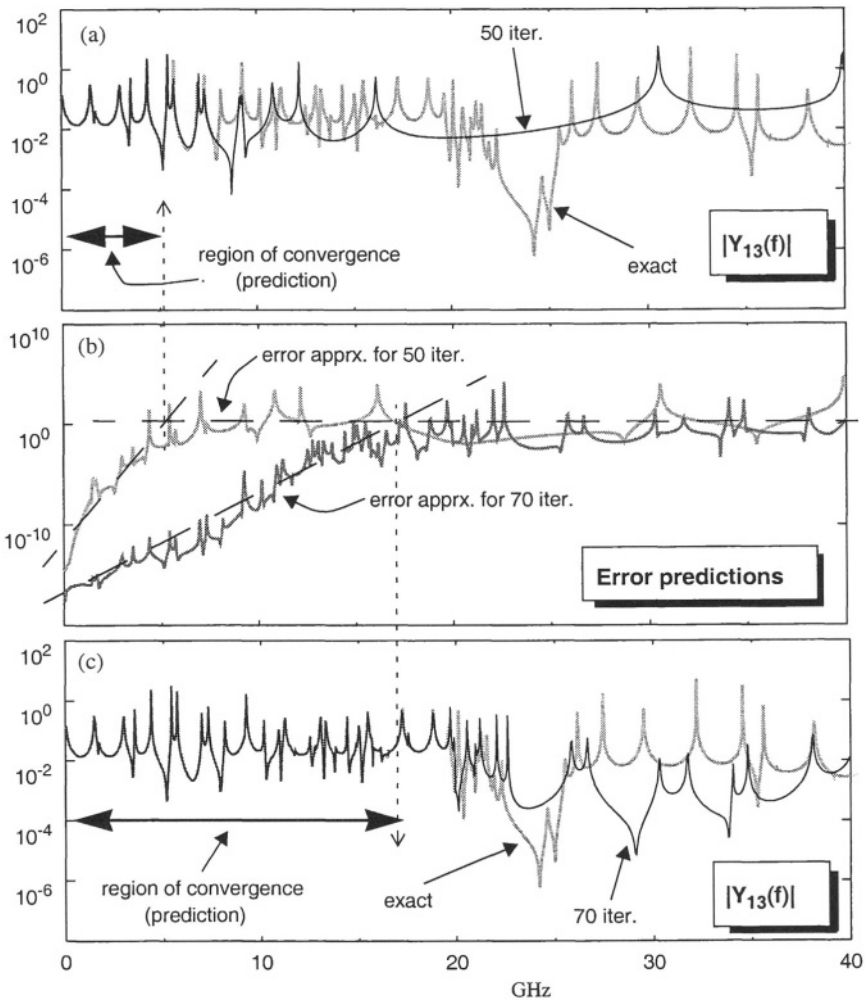


FIGURE 6.12 Demonstration of the convergence criteria based on the error measure.

References

[6.1] V. Raghavan, E. Bracken, and R. A. Rohrer, "AWESpice: A general tool for the accurate and efficient simulation of interconnect problems," in *Proc. Design Automation Conference,* June 1992.

[6.2] S. -Y. Kim, N. Gopal, and L. T. Pillage, "Time-domain macromodels for VLSI interconnect analysis," *IEEE Trans. on Computer-Aided Design,* vol 13, no. 10, pp. 1257-1270, Oct. 1994.

[6.3] R. Acar, M. Nakhla, P. Gunupudi, and E. Chiprout, "Passive interconnect reduction algorithm for distributed/measured networks," *IEEE Trans. on Circuits and Systems -- II,* vol. 17, no. 4, pp. 287-301, Apr. 2000.

[6.4] G. Strang, *Introduction to Linear Algebra,* Second Edition. Wellesley-Cambridge Press, June 1998.

[6.5] G. H. Golub and C. F. Van Loan, *Matrix Computations,* Third Edition. Baltimore: The Johns Hopkins University Press, 1996.

[6.6] C. -T. Chen, *Linear System Theory and Design,* Third Edition. New York: Oxford University Press, 1999.

[6.7] Y. Saad, *Iterative Methods For Sparse Linear Systems.* Boston, MA: PWS Pub. Co., 1996.

[6.8] C. D. Villemagne and R. E. Skelton, "Model reductions using a projection formulation," *Int. J. Control,* vol. 46, no. 6, pp. 2141-2169, 1987.

[6.9] H. M. Kim and R. R. Craig Jr., "Structural dynamics analysis using an unsymmetric block Lanczos algorithm," *Int. Journal for Numerical Methods In Engineering,* vol. 26, pp. 2305-18, 1988.

[6.10] E. J. Grimme, "Krylov projection methods for model reduction," Ph.D. dissertation, Univ. of Illinois, Urbana-Champaign, 1997.

[6.11] W. E. Arnoldi, "The principle of minimized iteration in the solution of the matrix eigenvalue problem," *Quart. Appl. Math.,* vol. 9, pp. 17-29, 1951.

[6.12] C. Lanczos, "An iteration method for the solution of the eigenvalue problem of linear differential and integral operators," *J. Res. Nat. Bur. Standards,* vol. 45. pp. 255-282,1950.

[6.13] D. L. Boley, "Krylov space methods on state-space control models," *Circuits Syst. Signal Process.* vol. 13, no. 6, pp. 733-758, 1994.

[6.14] J. I. Aliaga, D. L. Boley, R. W. Feund, and V. Hernandez, "A Lanczos-type method for multiple starting vectors," *Mathematics of Computation,* vol. 69, pp. 1577--1601, 2000.

[6.15] P. Feldmann and R. W. Freund, "Reduced-order modeling of large linear subcircuits via a block Lanczos algorithm," in *Proc. 32nd IEEE/ACM Design Automation Conf.,* pp. 474-479, Jun. 1995.

[6.16] R. W. Freund and P. Feldmann, "Reduced-order modeling of large linear passive multi-terminal circuits using matrix-Padé approximation," in *Proc. Design Automation and Test in Europe Conf.,* pp. 530-537, Feb. 1998.

[6.17] B. D. O. Anderson and S. Vongpanitlerd, *Network Analysis and Synthesis.* Enlewood Cliffs, NJ: Prentice-Hall Inc., 1973.

[6.18] R. A. Rohrer and H. Nosrati, "Passivity considerations in stability studies of numerical integration algorithms," *IEEE Trans. on Circuits and Systems,* vol. CAS-28, no. 9, pp. 857-866, Sep. 1981.

[6.19] A. Odabasioglu, M. Celik, and L. Pileggi, "PRIMA: Passive reduced-order interconnect macromodeling algorithm," in *Tech. Dig. 1997 IEEE/ACM Int. Conf. on Computer-Aided Design,* Nov. 1997.

[6.20] A. Odabasioglu, M. Celik, and L. Pileggi, "PRIMA: Passive reduced-order interconnect macromodeling algorithm," *IEEE Trans. on Computer-Aided Design of CAS,* vol. 18, no. 8, pp. 645-654, Aug. 1998.

[6.21] A. Odabasioglu, M. Celik, and L. Pileggi, "Practical considerations for passive reduction of RLC circuits," in *Proc. 36th IEEE/ACM Design Automation Conf.,* pp. 214-219, Jun. 1999.

[6.22] K. J. Kerns and A. T. Yang, "Preservation of passivity during RLC network reduction via split congruence transformations," in *Proc. 34th IEEE/ACM Design Automation Conf.,* pp. 34-39, Jun. 1997.

[6.23] K. J. Kerns and A. T. Yang, "Preservation of passivity during RLC network reduction via split congruence transformations," *IEEE Trans. on Computer-Aided Design of CAS,* vol. 17, no. 7, pp. 582-591, Jul. 1998.

[6.24] I. M. Elfadel and D. D. Ling, "A block rational Arnoldi algorithm for multipoint passive model-order reduction of multiport RLC networks," in *Tech. Dig. 1997 IEEE/ACM Int. Conf. on Computer-Aided Design*, pp. 66-71, Nov. 1997.

[6.25] R. W. Freund, "Passive reduced-order models for interconnect simulation and their computation via Krylov-subspace algorithms," in *Proc. 36th IEEE/ACM Design Automation Conf.*, Jun. 1999.

[6.26] A. C. Cangellaris, S. Pasha, J. L. Prince, and M. Celik, "A new discrete transmission line model for passive model order reduction and macromodeling of high-speed interconnections," *IEEE Trans. on Advanced Packaging*, vol. 22, pp. 356-364, Aug. 1999.

[6.27] M. Kamon, N. A. Marques, L. M. Silveira, and J. White, "Automatic generation of accurate circuit models of 3-D interconnect," *IEEE Trans. on Components, Packaging, and Manufacturing Tech. -- Part B*, vol. 21, pp. 225-240, Aug. 1998.

[6.28] A. E. Ruehli and A. C. Cangellaris, "Progress in the methodologies for the electrical modeling of interconnects and electronic packages," *Proc. IEEE*, vol. 89, no. 5, pp. 740-771, May 2001.

[6.29] A. C. Cangellaris and M. Celik, "Order reduction of high-speed interconnect electrical models: the issue of passivity," in *Proc. IC/Package Design Integration*, pp. 132 -137,1998.

[6.30] S. Pasha, M. Celik, A. C. Cangellaris, and J. L. Prince, "Passive SPICE-compatible models of dispersive interconnects," in *Proc. 49th Electronic Components and Technology Conference*, pp. 493-499,1999.

[6.31] A. E. Ruehli, "Equivalent circuit models for three-dimensional multiconductor systems," *IEEE Trans. Microwave Theory Tech.*, vol. MTT-22, Mar. 1974.

[6.32] R. W. Newcomb, *Linear Multiport Synthesis*. New York: McGraw-Hill, 1966.

[6.33] A. Ruhe, "The rational Krylov algorithm for nonsymmetric eigenvalue problems III: complex shifts for real matrices," *BIT*, vol. 34, pp. 165-176,1994.

[6.34] Q. Yu, J. M. L. Wang, E. S. Kuh, "Passive multipoint moment matching model order reduction algorithm on multiport distributed interconnect networks," *IEEE Transactions on Circuits and Systems -- I: Fund. Theory and Appl.* vol. 46, no. 1, pp. 140-160, Jan. 1999.

[6.35] A. Odabasioglu, "Passive reduced-order interconnect macromodeling algorithm," Ph. D. Thesis, Carnegie Mellon Univ., 1999.

[6.36] Y. Saad, "Krylov subspace methods for solving large unsymmetric linear systems," *Math. Comp.*, vol. 37, pp. 105-126, 1981.

[6.37] H. Levy, W. Scott, D. MacMillen, and J. White, "A rank-one update method for efficient processing of interconnect parasitics in timing analysis," in *Proc. 37th IEEE/ACM Design Automation Conf*, Jun. 2000.

[6.38] P. Feldmann and R. W. Freund, "Efficient linear circuit analysis by Padé approximations via the Lanczos process," *IEEE Trans. on Computer-Aided Design of CAS*, vol. 14, no. 5, pp. 639-649,May 1995.

[6.39] L. M. Silveira, M. Kamon, and J. White, "Efficient reduced-order modeling of frequency-dependent coupling inductances associated with 3-D interconnect structures," in *Proc. 32nd IEEE/ACM Design Automation Conf*, pp. 376-380, San Francisco, Jun. 1995.

[6.40] M. Celik and A. C. Cangellaris, "Simulation of multiconductor transmission lines using Krylov subspace order-reduction techniques," *IEEE Trans. on Computer-Aided Design*, vol. 16, no. 5, May 1997.

[6.41] J. E. Bracken, D. -K. Sun, Z. J. Cendes, "S-domain methods for simultaneous time and frequency characterization of electromagnetic devices," *IEEE Trans. on Microwave Theory and Techniques*, vol. 46, no. 9, pp. 1277-1290, Sep. 1998.

[6.42] J. Cullum, A. Ruehli, and T. Zhang, "A method for reduced-order modeling and simulation of large interconnect circuits and its application to PEEC models with retardation," *IEEE Trans. on Circuits and Systems -- II*, vol.47, Apr. 2000.

[6.43] K. Gallivan, E. Grimme, and P. Van Dooren, "Asymptotic waveform evaluation via Lanczos method," *Appl. Mat. Lett.*, vol. 7, no. 5, pp. 75-80, 1994.

[6.44] L. M. Silveira, M. Kamon, I. M. Elfadel, and J. White, "Coordinate-transformed Arnoldi for generating guaranteed stable reduced-order models for RLC circuits" in *Tech. Dig. 1996 IEEE/ACM Int. Conf. on Computer-Aided Design*, pp. 288-294, Nov. 1996.

[6.45] Z. Bai, R. D. Slone, W. T. Smith, and Q. Ye, "Error bound for reduced system model by Padé approximation via the Lanczos process," *IEEE Transactions on Computer-Aided Design of CAS*, vol. 18, no. 2, Feb. 1999.

CHAPTER 7

Interfacing with SPICE

In the previous chapter, we introduced techniques to find reduced order macromodels for large linear interconnect blocks. These macromodels are usually described in terms of frequency-domain matrix functions, which are not directly compatible with time-domain circuit simulation. In this chapter, we explain methods to convert these frequency-domain descriptions into time-domain equivalents. We will consider two alternatives: equivalent circuit synthesis and recursive convolution.

7.1 Multiport Interconnect Models

Consider the multiport shown in Figure 7.1. In the previous chapter, we saw that such a linear multiport circuit can be fully characterized with its *s*-domain admittance matrix, **Y**. We described different approximation methods in different forms. Here we will focus on two types of admittance matrix representations.

One of the representation types that we will consider in this chapter is in terms of poles and residues:

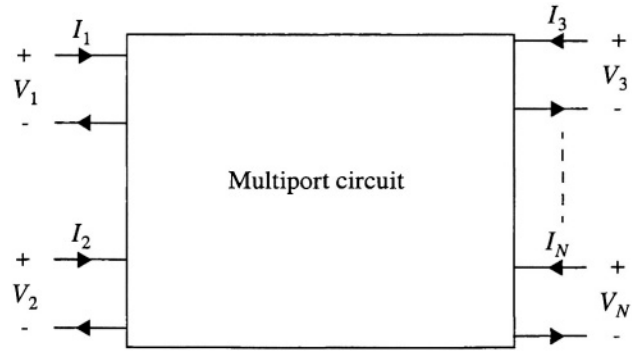


FIGURE 7.1 Circuit with N port

$$Y_{ij}(s) \approx \sum_{m=1}^{q_{ij}} \frac{k_{ijm}}{s - p_{ijm}} \qquad 1 \le i, j \le N \tag{7.1}$$

In general, a different set of poles can be used for each entry. Furthermore, the order for each entry can be different. Alternatively, a single set can be used for some or all of the entries. These cases can be collected in the three categories:

1. SISO: Each input-output pair is treated independently from the others producing a different set of poles for each entry. A straightforward application of AWE to each entry results in this type of admittance models.
2. SIMO: All the entries in a column of the admittance matrix share the same set of poles. For example, an AWE approximation can be obtained for Y_{jj} first and then the poles can be used for all other transfer functions in the same column. Similarly, a specific version of PRIMA can be used for each column (refer to Subsection 6.8.3).
3. MIMO: A single set of poles is used by all the entries in the admittance matrix. All projection methods, including PRIMA, produce this type of admittance models.

The other type of representation is in terms of reduced order circuit matrices:

$$\mathbf{Y}(s) = \mathbf{L}_q^T(\mathbf{G}_q + s\mathbf{C}_q)^{-1}\mathbf{B}_q \tag{7.2}$$

Such reduced order matrices can be obtained from PRIMA and other block Krylov techniques as we discussed in the last chapter, The matrix representation in (7.2) can easily be converted to a pole-residue form. It is also possible to obtain a matrix representation starting from the poles and residues as we explain later in this chapter.

There are basically two ways of combining multiport interconnect macromodels with other macromodels and (non)linear drivers and receivers in a circuit simulation environment:

1. *Synthesis:* An equivalent circuit is synthesized using basic circuit elements. Then any time-domain circuit simulator can be used.
2. *Recursive convolution:* This method requires a modification in the simulator in order to exploit the pole-residue information.

The remainder of this chapter presents these two techniques starting with the synthesis approach. But first we describe in detail how to obtain state-space and pole-residue representations of PRIMA-reduced models.

7.2 Post Processing of the PRIMA Models

Consider the general representation of a PRIMA-reduced model from Chapter 6:

$$\begin{aligned} \mathbf{G}_q\mathbf{x}_q + \mathbf{C}_q\frac{d\mathbf{x}_q}{dt} &= \mathbf{B}_q\mathbf{u}_p \\ \mathbf{i}_p &= \mathbf{L}_q^T\mathbf{x}_q \end{aligned} \tag{7.3}$$

and its admittance matrix

$$\mathbf{Y}_q(s) = \mathbf{L}_q^T(\mathbf{G}_q + s\mathbf{C}_q)^{-1}\mathbf{B}_q . \tag{7.4}$$

A. Reduction to State-Space Form With Cholesky Factorization

If $\mathbf{C}_q$ is invertible, we can simply multiply the first equation in (7.3) by $\mathbf{C}_q^{-1}$ to obtain a state-space form. Recall that for a passive model, the only requirement for the $\mathbf{C}_q$ matrix is that it be positive nonnegative. Thus, invertibility is not necessary. Most of the time, however, we expect it to be invertible or full rank.

Since $\mathbf{C}_q$ is symmetric, we use Cholesky factorization [7.1] rather than direct inversion or LU decomposition. The Cholesky factorization gives a lower triangular matrix $\mathbf{C}_L$ with positive diagonal entries such that

$$\mathbf{C}_q = \mathbf{C}_L \mathbf{C}_L^T \tag{7.5}$$

For a $q \times q$ matrix, the Cholesky algorithm involves only $q^3/3$ flops and it does not require any pivoting or reordering. In addition, it runs to completion only if the matrix is positive definite. Thus, a failure to complete the Cholesky factorization of the $\mathbf{C}_q$ matrix indicates a potential passivity problem in the reduced-order model. Such a passivity problem could be caused by numerical noise when generating the passive reduced-order model.

Defining $\mathbf{J} = \mathbf{C}_L^{-1}$, we can obtain a state-space form of the reduced order model by inserting (7.5) into (7.3)

$$\begin{aligned} \frac{d\mathbf{x}}{dt} &= \mathbf{A}_s \cdot \mathbf{x} + \mathbf{B}_s \cdot \mathbf{u}_p \\ \mathbf{i}_p &= \mathbf{C}_s^T \cdot \mathbf{x} \end{aligned} \tag{7.6}$$

where

$$\mathbf{A}_s = -\mathbf{J}\mathbf{G}_q\mathbf{J}^T \qquad \mathbf{B}_s = \mathbf{J}\mathbf{B}_q \qquad \mathbf{C}_s = \mathbf{J}\mathbf{L}_q \tag{7.7}$$

B. Reduction to State-Space Form With Eigendecomposition

The above method diagonalizes only one of the matrices, $\mathbf{C}_q$. However, we can (block-)diagonalize both of them using a similarity transformation. This process, which also gives the poles of the system, requires the solution of a generalized eigenvalue problem involving the reduced order MNA matrices $\mathbf{G}_q$ and $\mathbf{C}_q$. More precisely, we find an invertible transformation matrix $\mathbf{S}$ and a diagonal matrix Λ, such that

$$\mathbf{C}_q\mathbf{S} = -\mathbf{G}_q\mathbf{S}\Lambda \tag{7.8}$$

where $\mathbf{S}$ and Λ are, in general, complex matrices. Once the eigendecomposition in (7.8) is obtained, it can be inserted into (7.3) to block-diagonalize the system matrices.

We next explain the details of a robust approach for the solution of the eigenvalue problem in (7.8). Refer to [7.1] for various algorithms that are used in the solution of general eigenvalue problems. Source codes for some of these algorithms are available in the software package LAPACK [7.2].

Rather than separate $\mathbf{G}_q$ and $\mathbf{C}_q$ matrices, we work with the $\mathbf{A}_q = -\mathbf{G}_q^{-1}\mathbf{C}_q$ matrix, which we know from (6.107) that is a block upper Hessenberg matrix

$$\mathbf{A}_q = \begin{bmatrix} \mathbf{H}_{00} & \mathbf{H}_{01} & \cdots & \mathbf{H}_{0,k-2} & \mathbf{A}_{0,k-1} \\ \mathbf{H}_{10} & \mathbf{H}_{11} & \cdots & \mathbf{H}_{1,k-2} & \mathbf{A}_{1,k-1} \\ \mathbf{0} & \mathbf{H}_{21} & \cdots & \mathbf{H}_{2,k-2} & \mathbf{A}_{2,k-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{H}_{k-1,k-2} & \mathbf{A}_{k-1,k-1} \end{bmatrix} \tag{7.9}$$

The $\mathbf{H}_{ij}$ blocks in (7.9) are obtained directly from the block Arnoldi algorithm. The $\mathbf{A}_{ij}$ blocks are computed according to (6.107).

We first transform $\mathbf{A}_q$ to upper quasi-triangular form using the real Schur decomposition [7.1]. This corresponds to finding an orthogonal matrix $\mathbf{Q}$ and an upper quasi-triangular matrix $\mathbf{T}$ such that

$$\mathbf{Q}^T\mathbf{A}_q\mathbf{Q} = \mathbf{T} = \begin{bmatrix} \mathbf{T}_{11} & \mathbf{T}_{12} & \cdots & \mathbf{T}_{1m} \\ \mathbf{0} & \mathbf{T}_{22} & \cdots & \mathbf{T}_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{T}_{mm} \end{bmatrix} \tag{7.10}$$

where each $\mathbf{T}_{ii}$ is either a 1×1 or a 2×2 real matrix having complex conjugate eigenvalues.

The real Schur decomposition is achieved with a QR iteration algorithm:

$$
\begin{aligned}
&\mathbf{T}_0 = \mathbf{U}_0^T \mathbf{A}_q \mathbf{U}_0 \qquad \text{(Hessenberg reduction)} \\
&\text{for } k = 1, 2, \ldots \\
&\quad \mathbf{T}_{k-1} = \mathbf{U}_k \mathbf{R}_k \qquad \text{(QR factorization)} \\
&\quad \mathbf{T}_k = \mathbf{R}_k \mathbf{U}_k \\
&\text{end} \\
&\mathbf{Q} = \mathbf{U}_0 \mathbf{U}_1 \ldots \mathbf{U}_k \\
&\mathbf{T} = \mathbf{T}_k
\end{aligned}
\tag{7.11}
$$

In the first step, the matrix $\mathbf{A}_q$ is reduced to upper Hessenberg form, $\mathbf{T}_0$, by a sequence of Householder matrix operations or Givens rotations. Note that, in our case, $\mathbf{A}_q$ is already in the block upper Hessenberg form, therefore we can use Givens rotations to reduce it to actual Hessenberg form. It is more advantageous to use Given rotations because they zero out one element at a time, whereas Householder matrices zero out the entire column. The QR factorizations in (7.11) are performed until the $\mathbf{T}_k$'s converge to upper quasi-triangular form. The convergence rate can be accelerated by incorporating shifts [7.1]. The overall cost is approximately $30q^3$ flops, where q is the size of the matrix $\mathbf{A}_q$.

The algorithm in (7.11), however, does not generate the eigenvectors. It only yields the eigenvalues of $\mathbf{A}_q$ as 1×1 or 2×2 blocks on the diagonal of $\mathbf{T}$. The eigenvectors can be calculated as follows. Suppose that the kth diagonal element is a real eigenvalue,

$$
\mathbf{Q}^T \mathbf{A}_q \mathbf{Q} = \mathbf{T} = \begin{bmatrix} \mathbf{T}_A & \mathbf{u} & \ldots \\ \mathbf{0} & \lambda & \ldots \\ \mathbf{0} & \mathbf{0} & \ddots \end{bmatrix}
\tag{7.12}
$$

where $\mathbf{T}_A$ is a $(k-1)\times(k-1)$ upper quasi-triangular block and $\mathbf{u}$ is a vector of length $k-1$. Solve the linear system $(\mathbf{T}_A - \lambda \mathbf{I})\mathbf{w} = -\mathbf{u}$. The n-dimensional vector

$$
\mathbf{x} = \mathbf{Q} \begin{bmatrix} \mathbf{w} \\ 1 \\ \mathbf{0} \end{bmatrix}
\tag{7.13}
$$

is the corresponding right eigenvector,

$$\mathbf{A}_q\mathbf{x} = \mathbf{x}\lambda \tag{7.14}$$

The 2×2 diagonal blocks, which correspond to complex eigenvalue pairs, are handled similarly. For each 2×2 block, we obtain two real vectors. Combining the eigenvector equations for all eigenvalues, we obtain

$$\mathbf{A}_q\mathbf{S}_r = \mathbf{S}_r\mathbf{D} \tag{7.15}$$

where $\mathbf{D} = \text{diag}(\mathbf{T}_{11}, \mathbf{T}_{22}, \ldots, \mathbf{T}_{mm})$.

Now we turn our attention back to our reduced order system. With the $\mathbf{A}_q$ matrix representation, (7.3) becomes

$$\begin{aligned} \mathbf{x} &= \mathbf{A}_q\frac{d\mathbf{x}}{dt} + \mathbf{R}_q\mathbf{u}_p \\ \mathbf{i}_p &= \mathbf{L}_q^T\mathbf{x} \end{aligned} \tag{7.16}$$

where $\mathbf{R}_q = \mathbf{G}_q^{-1}\mathbf{B}_q$. Inserting (7.15) into (7.16) and after some manipulation we obtain

$$\begin{aligned} \frac{d\mathbf{x}}{dt} &= \mathbf{A}_s \cdot \mathbf{x} + \mathbf{B}_s \cdot \mathbf{u}_p \\ \mathbf{i}_p &= \mathbf{C}_s^T \cdot \mathbf{x} \end{aligned} \tag{7.17}$$

The reduced-order state matrix $\mathbf{A}_s$ is the inverse of the block-diagonal matrix $\mathbf{D}$,

$$\mathbf{A}_s = \text{diag}(\mathbf{A}_{11}, \mathbf{A}_{22}, \ldots, \mathbf{A}_{mm}) \tag{7.18}$$

where each $\mathbf{A}_{ii}$ is either a 1×1 or a 2×2 block

$$\mathbf{A}_{ii} = \mathbf{T}_{ii}^{-1} \qquad 1 \le i \le m \tag{7.19}$$

The other two state-space matrices are defined as

$$\mathbf{B}_s = \mathbf{A}_s\mathbf{S}_r^{-1}\mathbf{R}_q \quad \text{and} \quad \mathbf{C}_s = \mathbf{S}_r^T\mathbf{L}_q \tag{7.20}$$

If desired, the reduced order model can also be represented in terms of the poles and residues of its matrix admittance function. To achieve this, we need one more similarity transformation

$$\mathbf{A}_s = \mathbf{J}\Lambda\mathbf{J}^{-1} \tag{7.21}$$

which reduces the block diagonal $\mathbf{A}_s$ to the diagonal matrix Λ. This time, Λ and the block-diagonal transformation matrix $\mathbf{J}$ may have complex elements. Assuming the same form of partitioning with $\mathbf{A}_s$, we construct Λ and $\mathbf{J}$ as follows:

If $\mathbf{A}_{ii}$ is a 1×1 block, set $\Lambda_{ii} = \mathbf{A}_{ii}$ and $\mathbf{J}_{ii} = 1$;
Otherwise, $\mathbf{A}_{ii}$ must be a 2×2 block of the form

$$\mathbf{A}_{ii} = \begin{bmatrix} \alpha & \beta \\ -\beta & \alpha \end{bmatrix}, \tag{7.22}$$

$$\text{set } \Lambda_{ii} = \begin{bmatrix} \alpha + j\beta & 0 \\ 0 & \alpha - j\beta \end{bmatrix} \text{ and } \mathbf{J}_{ii} = \begin{bmatrix} 1 & 1 \\ j & -j \end{bmatrix}. \tag{7.23}$$

Inserting (7.21) into (7.17) and after some matrix operations, we obtain

$$\begin{aligned} \frac{d\mathbf{x}}{dt} &= \Lambda\mathbf{x} + \mathbf{J}^{-1}\mathbf{B}_s\mathbf{u}_p \\ \mathbf{i}_p &= \mathbf{C}_s^T\mathbf{J}\mathbf{x} \end{aligned} \tag{7.24}$$

From (7.24) it follows that

$$\mathbf{Y}_q(s) = \mathbf{C}_s^T\mathbf{J}(s\mathbf{I} - \Lambda)^{-1}\mathbf{J}^{-1}\mathbf{B}_s \tag{7.25}$$

Thus

$$Y_{ij}(s) = \sum_{k=1}^{q} \frac{\mu_{ik} \cdot \nu_{kj}}{s - \lambda_k} \qquad 1 \le i, j \le N \tag{7.26}$$

where λ_k is the kth diagonal element of Λ, μ_{ik} is the (i, k)th element of the $N \times q$ matrix $\mathbf{C}_s^T\mathbf{J}$, ν_{kj} is the (k, j) th element of the $q \times N$ matrix $\mathbf{J}^{-1}\mathbf{B}_s$, and N is the size of the $\mathbf{Y}_q(s)$ matrix. Note that the poles, λ_k's, are common to all elements of $\mathbf{Y}_q(s)$. Thus

$$\mathbf{Y}_q(s) = \sum_{i=1}^{q} \frac{\mathbf{K}_i}{s - p_i} \tag{7.27}$$

where $\mathbf{K}_i$ is the $N \times N$ residue matrix corresponding the ith pole of the system.

Having seen how to post-process PRIMA-reduced models, we now return to the general problem: state-space realization from poles and residues.

7.3 State-Space Realization from Poles and Residues

There are two steps involved in generating an equivalent circuit for multiport macromodels. First a state-space representation is obtained, which is also known as state space realization. The second step is to synthesize an equivalent circuit. In this section we explain the realization part. In the following section, we describe the synthesis part.

The problem of state-space realization is defined as follows [7.3]: Given an s-domain matrix function $\mathbf{Y}(s)$, find the matrices $\{\mathbf{A}, \mathbf{B}, \mathbf{C}\}$ in

$$\begin{aligned} \frac{d\mathbf{x}}{dt} &= \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \\ \mathbf{i} &= \mathbf{C}^T\mathbf{x} \end{aligned} \tag{7.28}$$

such that

$$\mathbf{Y}(s) = \mathbf{C}^T(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} \tag{7.29}$$

The matrix function to be realized, $\mathbf{Y}(s)$, can be expressed in terms of matrices, which is the case in the projection-based order reduction methods such as PRIMA. Construction of the state-space descriptions from PRIMA was explained in the previous section.

Alternatively, $\mathbf{Y}(s)$ can be specified in terms of poles and residues. We now discuss this case.

We start with the simplest case. Consider a linear one-port and assume that its admittance is approximated with a one-pole model:

$$Y(s) = \frac{k}{s-p} \tag{7.30}$$

where the pole p *is* a negative real number with the associated residue k. A state-space realization for this admittance is as follows:

$$\begin{aligned} \frac{dx}{dt} &= px + v(t) \\ i(t) &= kx \end{aligned} \tag{7.31}$$

where $v(t)$ and $i(t)$ represent the port voltage and current, respectively, and $x(t)$ is the state introduced.

Let us now assume that the admittance of the one-port is a q-pole approximation:

$$Y(s) = \frac{k_1}{s-p_1} + \frac{k_2}{s-p_2} + \ldots + \frac{k_q}{s-p_q} \tag{7.32}$$

where all the poles are negative real numbers. In this case, a state-space realization is achieved by introducing q state variables $x_i(t)$, $1 \le i \le q$ such that

$$X_i = \frac{V}{s-p_i} \qquad 1 \le i \le q \tag{7.33}$$

where V and X_i are the Laplace transforms of $v(t)$ and $x_i(t)$, respectively. The state-space equations are then written as

$$\begin{aligned} \frac{d\mathbf{x}}{dt} &= \begin{bmatrix} p_1 & 0 & \ldots & 0 \\ 0 & p_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & p_q \end{bmatrix} \mathbf{x} + \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} v(t) \\ i(t) &= \begin{bmatrix} k_1 & k_2 & \ldots & k_q \end{bmatrix} \mathbf{x} \end{aligned} \tag{7.34}$$

or in a more compact form

$$\begin{aligned} \frac{d\mathbf{x}}{dt} &= \mathbf{P}\mathbf{x} + \mathbf{b}v(t) \\ i(t) &= \mathbf{k}^T\mathbf{x} \end{aligned} \tag{7.35}$$

In the presence of complex poles, the representation becomes slightly more complicated. To avoid complex numbers in the state matrix, a similarity transformation is used. We now explain this with a simple case. Since the complex poles appear in conjugate pairs, we assume that the admittance is of the form

$$Y(s) = \frac{k}{s-p} + \frac{\bar{k}}{s-\bar{p}} \tag{7.36}$$

where $p = a + bj, \quad k = c + dj$.

A state-space realization of (7.36) is in the form of (7.35), where

$$\mathbf{P} = \begin{bmatrix} a+bj & 0 \\ 0 & a-bj \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \qquad \mathbf{k}^T = \begin{bmatrix} c+dj & c-dj \end{bmatrix} \tag{7.37}$$

It can be shown that the two realizations shown below are equivalent as long as the similarity transformation matrix $\mathbf{J}$ is non-singular:

$$\begin{aligned} \frac{d\mathbf{x}}{dt} &= \mathbf{A}\mathbf{x} + \mathbf{b}v \\ i &= \mathbf{c}^T\mathbf{x} \end{aligned} \quad \Leftrightarrow \quad \begin{aligned} \frac{d\mathbf{x}}{dt} &= \mathbf{J}\mathbf{A}\mathbf{J}^{-1}\mathbf{x} + \mathbf{J}\mathbf{b}v \\ i &= \mathbf{c}^T\mathbf{J}^{-1}\mathbf{x} \end{aligned} \tag{7.38}$$

Choosing $\mathbf{J}$ as

$$\mathbf{J} = \begin{bmatrix} 1 & 1 \\ j & -j \end{bmatrix} \tag{7.39}$$

and applying the transformations $\mathbf{J}\mathbf{A}\mathbf{J}^{-1} \rightarrow \mathbf{A}$, $\mathbf{J}\mathbf{b} \rightarrow \mathbf{b}$, and $\mathbf{c}^T\mathbf{J}^{-1} \rightarrow \mathbf{c}^T$, we get

$$\mathbf{P} = \begin{bmatrix} a & -b \\ b & a \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} 2 \\ 0 \end{bmatrix} \qquad \mathbf{k}^T = \begin{bmatrix} c & d \end{bmatrix} \tag{7.40}$$

Next we extend these results to linear multiports. We will consider three cases: SISO, SIMO, and MIMO as defined in Section 7.1.

First consider the SIMO case, that is, take only one column, say *j*th, in the admittance matrix $\mathbf{Y}(s)$ and assume that all of the entries in the column share the same set of poles,

$$Y_{ij}(s) = \frac{k_{ij1}}{s-p_{j1}} + \frac{k_{ij2}}{s-p_{j2}} + \ldots + \frac{k_{ijq}}{s-p_{jq}} \qquad 1 \le i \le N \tag{7.41}$$

In terms of input voltage and output currents, (7.41) is equivalent to

$$\begin{bmatrix} I_1 \\ I_2 \\ \vdots \\ I_N \end{bmatrix} = \begin{bmatrix} k_{1j1} & k_{1j2} & \ldots & k_{1jq} \\ k_{2j1} & k_{2j2} & \ldots & k_{2jq} \\ \vdots & \vdots & \ddots & \vdots \\ k_{Nj1} & k_{Nj2} & \ldots & k_{Njq} \end{bmatrix} \begin{bmatrix} 1/(s-p_{j1}) \\ 1/(s-p_{j2}) \\ \vdots \\ 1/(s-p_{jq}) \end{bmatrix} V_j \tag{7.42}$$

It can be shown that the following is a realization for (7.42):

$$\begin{aligned} \frac{d\mathbf{x}}{dt} &= \mathbf{P}_j\mathbf{x} + \mathbf{b}_j v_j \\ \mathbf{i} &= \mathbf{K}_j\mathbf{x} \end{aligned} \tag{7.43}$$

where $\mathbf{K}_j$ is the residue matrix in (7.42) and, $\mathbf{P}_j$ and $\mathbf{b}$ are as defined in (7.34) and (7.35), but expressed in terms of the poles used in (7.41). The realization of the entire $\mathbf{Y}(s)$ matrix is obtained by combining the state equations given in (7.43) for all inputs:

$$\begin{aligned} \frac{d\mathbf{x}}{dt} &= \begin{bmatrix} \mathbf{P}_1 & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \mathbf{P}_2 & \ldots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{P}_N \end{bmatrix} \mathbf{x} + \begin{bmatrix} \mathbf{b}_1 & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \mathbf{b}_2 & \ldots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{b}_N \end{bmatrix} \mathbf{v} \\ \mathbf{i} &= \begin{bmatrix} \mathbf{K}_1 & \ldots & \mathbf{K}_N \end{bmatrix} \mathbf{x} \end{aligned} \tag{7.44}$$

We should point out that the sizes of $\mathbf{P}_i$'s are not required to be the same, that is, the number of poles can be different for each SIMO system. Also note that all $\mathbf{P}_i$'s can be the same, corresponding to the realization of an admittance matrix with a common set of poles for all entries, i.e., the MIMO case.

Finally, we discuss the SISO case in which each entry has a different set of poles, i.e.,

$$Y_{ij}(s) = \frac{k_{ij1}}{s - p_{ij1}} + \frac{k_{ij2}}{s - p_{ij2}} + \ldots + \frac{k_{ijq}}{s - p_{ijq}} \tag{7.45}$$

A realization for each entry is similar to the one obtained for the SISO system given in (7.35):

$$\begin{aligned} \frac{d\mathbf{x}}{dt} &= \mathbf{P}_{ij}\mathbf{x} + \mathbf{b}_{ij}v_j \\ i_i &= \mathbf{k}_{ij}\mathbf{x} \end{aligned} \tag{7.46}$$

By combining all the input output pairs -- their total number is N^2 -- a realization for the whole admittance matrix can be obtained.

As an example, consider the admittance matrix of a two-port:

$$\mathbf{Y}(s) = \begin{bmatrix} Y_{11}(s) & Y_{12}(s) \\ Y_{21}(s) & Y_{22}(s) \end{bmatrix} \tag{7.47}$$

and assume each entry has a state space realization of $\{\mathbf{P}_{ij}, \mathbf{b}_{ij}, \mathbf{k}_{ij}\}$ as defined in (7.46). Then a realization for $\mathbf{Y}(s)$ is given by

$$\begin{aligned} \frac{d\mathbf{x}}{dt} &= \begin{bmatrix} \mathbf{P}_{11} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{P}_{12} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{P}_{21} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{P}_{22} \end{bmatrix} \mathbf{x} + \begin{bmatrix} \mathbf{b}_{11} & \mathbf{0} \\ \mathbf{0} & \mathbf{b}_{12} \\ \mathbf{b}_{21} & \mathbf{0} \\ \mathbf{0} & \mathbf{b}_{22} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \\ \begin{bmatrix} i_1 \\ i_2 \end{bmatrix} &= \begin{bmatrix} \mathbf{k}_{11} & \mathbf{k}_{12} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{k}_{21} & \mathbf{k}_{22} \end{bmatrix} \mathbf{x} \end{aligned} \tag{7.48}$$

Note that the size of the state matrix for the SIMO and MIMO cases is Nq while it is N^2q for the SISO case. Assuming that the number of poles, q, is comparable in all cases, it is more efficient to use a SIMO or MIMO based technique. Also, we recognize that the MIMO approach does not have an advantage over SIMO with respect to the size of the state matrix.

7.4 Synthesis of State-Space Realizations

We now synthesize equivalent circuits for the state-space descriptions. We again start with the simplest case. Take the Laplace transform of (7.31) which is the realization of $Y(s) = k/(s-p)$, one-pole admittance model of a one-port. After some manipulation we obtain

$$\begin{aligned} sV_x - pV_x &= V \\ I &= kV_x \end{aligned} \tag{7.49}$$

where V_x is the Laplace transform of $x(t)$. The system given in (7.49) can be synthesized with simple circuit elements as shown in Figure 7.2.

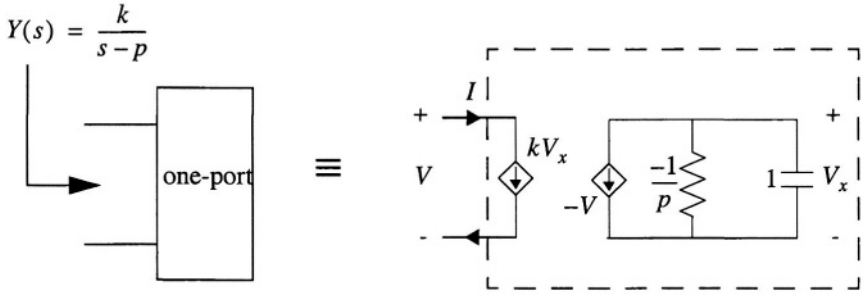


FIGURE 7.2 One-pole admittance and its realization.

We now consider the general case: multiport multipole admittance models. Assume that the admittance matrix has a state-space realization in the form

$$\begin{aligned} \frac{d\mathbf{x}}{dt} &= \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}(t) \\ \mathbf{i}(t) &= \mathbf{C}^T\mathbf{x} \end{aligned} \tag{7.50}$$

The above system can be implemented using basic linear circuit elements such as capacitors, resistances, and voltage controlled current sources. In the equivalent circuit, an internal node is created for each entry of the state vector. Consider a row from the first matrix equation in (7.50). In the Laplace domain it becomes

$$-a_{i1}X_1 - a_{i1}X_2 - \ldots + (s - a_{ii})X_i - \ldots - a_{iq}X_q = b_{i1}U_1 + \ldots + b_{iN}U_N \qquad (7.51)$$

The $(s - a_{ii})X_i$ term is realized with a parallel combination of a capacitor of a unity value and a resistance with a value of $-1/a_{ii}$. All the other terms on the left hand side are represented as voltage controlled current sources. All the terms on the right hand side are also current sources which are controlled by the port voltages, U_i. The equivalent circuit is shown in Figure 7.3.

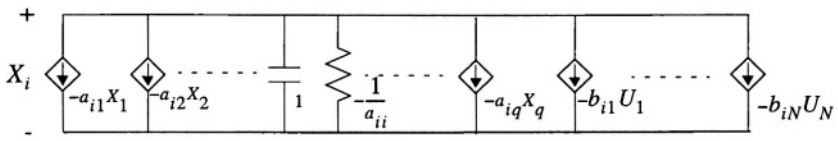


FIGURE 7.3 Synthesis of a row in state-space realization.

Now consider a row from the second matrix equation in (7.50):

$$I_k = c_{1k}X_1 + \ldots + c_{qk}X_q \qquad (7.52)$$

which can be synthesized as voltage controlled current sources at the ports as shown in Figure 7.4. Figure 7.5 shows the complete circuit which synthesizes (7.50). Each of the blocks in the middle corresponds to a state variable.

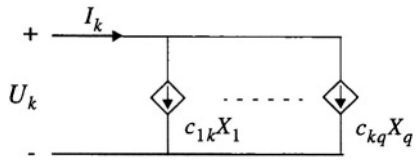


FIGURE 7.4 Synthesis of a port in state-space realization.

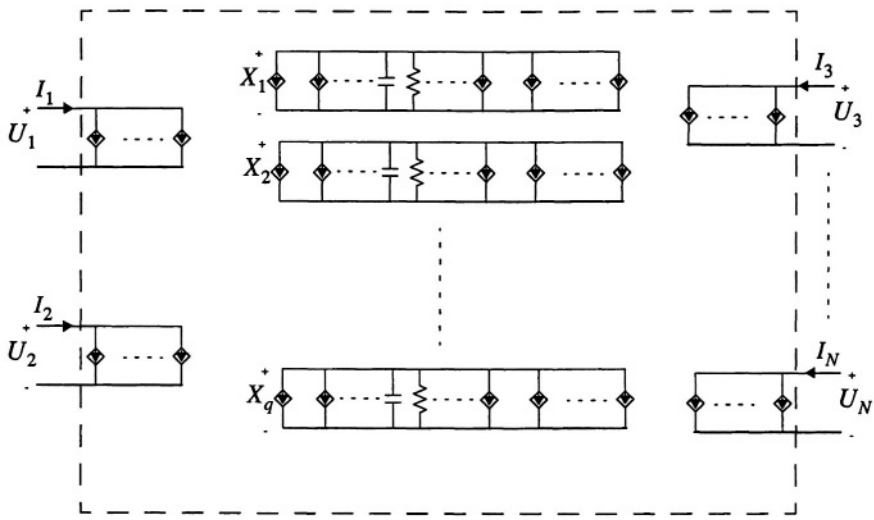


FIGURE 7.5 Synthesis of state-space realization in terms of blocks of Figure 7.3 and Figure 7.4.

7.5 *Recursive Convolution*

Another way of converting a frequency-domain description to a time-domain model is through convolution, which, in general, has a quadratic runtime cost. But if the frequency-domain descriptions are in terms of poles and residues we can exploit this fact and evaluate the convolutions in a recursive manner so that the computational cost is constant regardless of the time. In this section we present an efficient recursive convolution method which is also very easy to implement.

7.5.1 Numerical Convolution versus Recursive Convolution

Consider a one-port with an *s*-domain admittance model

$$I(s) = Y(s)V(s) \tag{7.53}$$

In the time domain, its branch equation is given by the convolution

$$\begin{aligned} i(t) &= y(t)*v(t) \\ &= \int_{-\infty}^{t} y(t-\tau)v(\tau)d\tau \end{aligned} \tag{7.54}$$

Now suppose that a voltage source is connected between the terminals of the one-port. We seek to find the values of $i(t)$ at discrete time points $t = t_i$. Assume that we can approximate $v(t)$ and $y(t)$ as piecewise constant waveforms. Namely,

$$\left.\begin{aligned} v(t) &= v(t_i) \\ y(t) &= y(t_i) \end{aligned}\right\} \quad \text{for} \quad t_{i-1} < t \leq t_i \tag{7.55}$$

Assuming that all the past values of $v(t)$ are available, the value of $i(t_k)$ can be computed by discretizing the integral given in (7.54):

$$i(t_k) = \sum_{i=0}^{k} y(t_{k-i})v(t_i)\Delta_i \tag{7.56}$$

where we have assumed that $v(t) = 0$ for $t < 0$.

Two obvious drawbacks of the numerical convolution method are immediately recognized from (7.56). First, all the past values of the waveforms must be stored, and more importantly it has a complexity of $O(T^2)$. In addition, we have to compute the impulse response, $y(t)$, from $Y(s)$. For this, we can use either numerical inverse Laplace transformation [7.4] or inverse fast Fourier transformation type approaches [7.5]. Both of these transformations are often subject to some numerical problems.

Now suppose that the poles and residues of the admittance function are known. For the sake of simplicity consider a one-pole model

$$Y(s) = \frac{k}{s-p} \tag{7.57}$$

whose inverse Laplace transform is given by the closed form expression

$$y(t) = ke^{pt}. \tag{7.58}$$

Thus the convolution integration in (7.54) becomes

$$i(t) = \int_0^t k e^{p(t-\tau)} v(\tau) d\tau \tag{7.59}$$

Dividing the integration into two parts, the value of the current at time t_k can be expressed as

$$\begin{aligned} i(t_k) &= \int_0^{t_{k-1}} k e^{p(t_k-\tau)} v(\tau) d\tau + \int_{t_{k-1}}^{t_k} k e^{p(t_k-\tau)} v(\tau) d\tau \\ &= e^{p\Delta_k} \int_0^{t_{k-1}} k e^{p(t_{k-1}-\tau)} v(\tau) d\tau + \int_{t_{k-1}}^{t_k} k e^{p(t_k-\tau)} v(\tau) d\tau \end{aligned} \tag{7.60}$$

It is easily recognized that the first integral in the second equation in (7.60) is nothing but $i(t_{k-1})$. Evaluating the second integral in (7.60) with a piecewise constant voltage waveform assumption, we obtain the following recursion to compute $i(t_k)$:

$$i(t_k) = e^{p\Delta_k} i(t_{k-1}) + v(t_k) \cdot \frac{k(1 - e^{p\Delta_k})}{p} \tag{7.61}$$

It is apparent from (7.61) that we only need the most recent values of the waveforms. Furthermore, the number of operations at each time step is fixed which results in a computational complexity of $O(T)$.

In the above discussion we have assumed that the value of $v(t)$ is known at the current time point. This assumption is true only if a voltage source is connected to the terminals of the one-port. Otherwise, and in general, the value of $v(t_k)$ is unknown as well, and has to be calculated simultaneously with $i(t_k)$ and other circuit variables. Hence (7.61) can be interpreted as an equivalent conductance and an equivalent current source,

$$i(t_k) = v(t_k) G_{eq}(t_k) + I_{eq}(t_k) \tag{7.62}$$

which is illustrated in Figure 7.6.

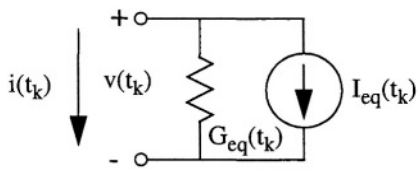


FIGURE 7.6 Time-domain model of an admittance in recursive convolution.

Recursive convolution was first used in [7.6] for power transmission line simulations. Recently, it has become popular again, this time for digital interconnect simulation. After the development of model order reduction techniques, several recursive convolution methods have been proposed in order to use these reduced order models in nonlinear simulators [7.7] [7.8] [7.9] [7.10]. Next, we present an efficient and simple implementation of recursive convolution.

7.5.2 A Simple Implementation of Recursive Convolution

In the case of a q-pole admittance model for the one-port,

$$Y(s) = \sum_{i=1}^{q} \frac{k_i}{s - p_i} \tag{7.63}$$

the convolution in (7.59) becomes

$$i(t) = \sum_{i=1}^{q} k_i x_i(t) \tag{7.64}$$

where the state $x_i(t)$ corresponds to the pole p_i and is defined as

$$x_i(t) = \int_0^t e^{p_i(t-\tau)} v(\tau) d\tau \tag{7.65}$$

Using a similar approach as in the last subsection, we can manipulate the above integration to obtain

$$x_i(t_k) = e^{p_i\Delta_k} x_i(t_{k-1}) + \int_{t_{k-1}}^{t_k} e^{p_i(t_k-\tau)} v(\tau)\,d\tau \tag{7.66}$$

Since we do not know the exact shape of $v(t)$ between t_{k-1} and t_k, the integral in (7.66) cannot be computed explicitly. In the introductory implementation described in the previous subsection we assumed that the voltage waveform is piecewise constant so that we could evaluate the integral analytically. A more acceptable waveshape is piecewise linear as shown in Figure 7.7. Thus, the voltage between t_{k-1} and t_k is expressed as follows:

$$v(t) = v(t_{k-1}) + \frac{t - t_{k-1}}{\Delta_k} \cdot (v(t_k) - v(t_{k-1})) \tag{7.67}$$

Using the piecewise linear assumption in (7.67), the integral in (7.66) can be evaluated explicitly:

$$x_i(t_k) = e^{p_i\Delta_k} x_i(t_{k-1}) + \frac{e^{p_i\Delta_k}(p_i\Delta_k - 1) + 1}{p_i^2\Delta_k} \cdot v(t_{k-1}) + \frac{e^{p_i\Delta_k} - 1 - p_i\Delta_k}{p_i^2\Delta_k} \cdot v(t_k) \tag{7.68}$$

We can rewrite the expression in (7.68) as

$$x_i(t_k) = \zeta_i(\Delta_k) \cdot x_i(t_{k-1}) + \eta_i(\Delta_k) \cdot v(t_{k-1}) + \theta_i(\Delta_k) \cdot v(t_k) \tag{7.69}$$

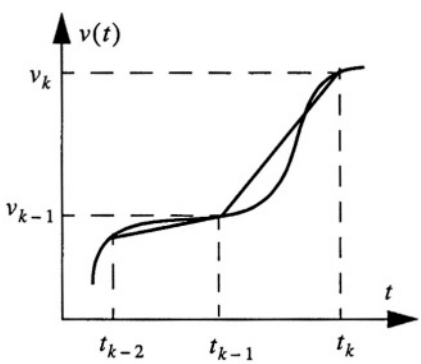


FIGURE 7.7 Piecewise linear assumption of a waveform.

with the parameters ζ, η and θ defined as

$$\begin{aligned}\zeta_i(\Delta_k) &= e^{p_i\Delta_k} \\ \eta_i(\Delta_k) &= \frac{e^{p_i\Delta_k}(p_i\Delta_k - 1) + 1}{p_i^2\Delta_k} \\ \theta_i(\Delta_k) &= \frac{e^{p_i\Delta_k} - 1 - p_i\Delta_k}{p_i^2\Delta_k}\end{aligned} \tag{7.70}$$

These parameters depend only on the poles and the time step. They remain constant when the time step is fixed for consecutive time points.

The current at time point t_k can be expressed as

$$i(t_k) = \sum_{i=1}^{q} k_i x_i(t_k) = I_{eq}(t_k) + G_{eq}(t_k) \cdot v(t_k) \tag{7.71}$$

where the equivalent current source and the equivalent conductance are defined as

$$\begin{aligned}G_{eq}(t_k) &= \sum_{i=1}^{q} k_i \cdot \theta_i \\ I_{eq}(t_k) &= \left(\sum_{i=1}^{q} k_i \cdot \eta_i\right) v(t_{k-1}) + \sum_{i=1}^{q} k_i \zeta_i x_i(t_{k-1})\end{aligned} \tag{7.72}$$

Once the voltage at time point t_k is known from the solution of the overall circuit, the states should be updated using (7.69) to perform the next time point calculation.

Complex poles are handled slightly differently. In this case, each conjugate pair owns two states coupled to each other. Consider a pair of conjugate poles, p_i and p_{i+1}, and their residues:

$$p_i = a + bj \qquad p_{i+1} = a - bj \qquad k_i = c + dj \qquad k_{i+1} = c - dj \tag{7.73}$$

One can show that the corresponding states are updated as follows [7.10]:

$$\begin{bmatrix} x_i^1(t_k) \\ x_i^2(t_k) \end{bmatrix} = \begin{bmatrix} e^{a\Delta_k}\cos b\Delta_k & -e^{a\Delta_k}\sin b\Delta_k \\ e^{a\Delta_k}\sin b\Delta_k & e^{a\Delta_k}\cos b\Delta_k \end{bmatrix} \begin{bmatrix} x_i^1(t_{k-1}) \\ x_i^2(t_{k-1}) \end{bmatrix} + \frac{2}{\Delta_k(a^2+b^2)^2}\begin{bmatrix} \alpha(\Delta_k) \\ \beta(\Delta_k) \end{bmatrix} v(t_{k-1}) + \frac{2}{\Delta_k(a^2+b^2)^2}\begin{bmatrix} \gamma(\Delta_k) \\ \delta(\Delta_k) \end{bmatrix} v(t_k), \tag{7.74}$$

where the parameters α, β, γ, and δ are defined as

$$\begin{aligned} \alpha(\Delta_k) &= [-(a^2-b^2-\Delta_k a(a^2+b^2))\cos b\Delta_k - (2ab-\Delta_k b(a^2+b^2))\sin b\Delta_k]e^{a\Delta_k} + (a^2-b^2) \\ \beta(\Delta_k) &= [-(2ab-\Delta_k b(a^2+b^2))\cos b\Delta_k + (a^2-b^2-\Delta_k a(a^2+b^2))\sin b\Delta_k]e^{a\Delta_k} + 2ab \\ \gamma(\Delta_k) &= b^2-a^2-\Delta_k a(a^2+b^2) + e^{a\Delta_k}[(a^2-b^2)\cos b\Delta_k + 2ab\sin b\Delta_k] \\ \delta(\Delta_k) &= -2ab-\Delta_k b(a^2+b^2) + e^{a\Delta_k}[2ab\cos\Delta_k - (a^2-b^2)\sin b\Delta_k] \end{aligned} \tag{7.75}$$

A. Extension to Multiports

We will explain the multiport case with a two-port example. Consider the frequency-domain description of a two-port:

$$\begin{bmatrix} I_1(s) \\ I_2(s) \end{bmatrix} = \begin{bmatrix} Y_{11}(s) & Y_{12}(s) \\ Y_{21}(s) & Y_{22}(s) \end{bmatrix} \begin{bmatrix} V_1(s) \\ V_2(s) \end{bmatrix} \tag{7.76}$$

where the four admittance terms are given by

$$\begin{aligned} Y_{11}(s) &= \sum_{i=1}^{q} \frac{k_{11i}}{s-p_{11i}}, \quad Y_{12}(s) = \sum_{i=1}^{q} \frac{k_{12i}}{s-p_{12i}} \\ Y_{21}(s) &= \sum_{i=1}^{q} \frac{k_{21i}}{s-p_{21i}}, \quad Y_{22}(s) = \sum_{i=1}^{q} \frac{k_{22i}}{s-p_{22i}} \end{aligned} \tag{7.77}$$

Our goal is to calculate the discrete time-domain model at each time point:

$$\begin{bmatrix} i_1(t_k) \\ i_2(t_k) \end{bmatrix} = \begin{bmatrix} G_{11}(t_k) & G_{12}(t_k) \\ G_{21}(t_k) & G_{22}(t_k) \end{bmatrix} \begin{bmatrix} v_1(t_k) \\ v_2(t_k) \end{bmatrix} + \begin{bmatrix} I_{1eq}(t_k) \\ I_{2eq}(t_k) \end{bmatrix} \tag{7.78}$$

From the circuit analysis point of view, the equivalent circuit of (7.78) is shown in Figure 7.8. The discrete time domain model for a two-port at time point t_k consists of an equivalent conductance, an equivalent current source, and an equivalent VCCS at each port. For a general multi-port, the discrete time-domain model will have the same topology, except that there will be more voltage-controlled current sources.

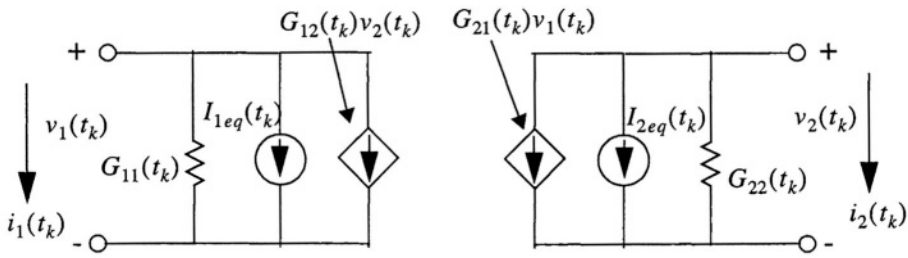


FIGURE 7.8 The time-domain equivalent of a two-port.

During the simulation process, at each time point, a discrete time domain model similar to the one shown in (7.78), is constructed first, based on a reduced-order model expressed in the frequency domain, as shown in (7.76). Then the model in (7.78) is combined with the models connected to its ports via a circuit equation formulation scheme such as the Modified Nodal Analysis (MNA). The solution of the nonlinear MNA equations is obtained using a numerical integration approximation for the capacitors and inductors that are not within the N-port(s), and a nonlinear iteration method such as Newton-Raphson [7.11].

Upon choosing the appropriate time step, the parameters ζ, η, θ for each real pole are calculated based on the formulae in (7.70). The equivalent conductance can then be calculated as:

$$G_{11}(t_k) = \sum_{i=1}^{q} k_{11i} \cdot \theta_i^{(11)}(\Delta t_k), \quad G_{12}(t_k) = \sum_{i=1}^{q} k_{12i} \cdot \theta_i^{(12)}(\Delta t_k)$$
$$G_{21}(t_k) = \sum_{i=1}^{q} k_{21i} \cdot \theta_i^{(21)}(\Delta t_k), \quad G_{22}(t_k) = \sum_{i=1}^{q} k_{22i} \cdot \theta_i^{(22)}(\Delta t_k) \tag{7.79}$$

The number of states to update is $4q$. These states can be stored in a single vector $\mathbf{x}$:

$$\mathbf{x} = \left[x_1^{(11)} \ldots x_q^{(11)}\ x_1^{(21)} \ldots x_q^{(21)}\ x_1^{(12)} \ldots x_q^{(12)}\ x_1^{(22)} \ldots x_q^{(22)}\right] \tag{7.80}$$

At each time point, if the time step is unchanged, only the equivalent current sources have to be updated. This procedure can be carried out as shown in Figure 7.9. Note that all the floating-point operations involved are SAXPY ($y \leftarrow a \cdot x + y$) type. For a modern computer architecture this is the simplest floating point operation and therefore, it can be carried out with the utmost efficiency.

B. Local Truncation Error (LTE) of the Algorithm

Following the common assumption of LTE estimation, we assume that the exact value of the state at time point t_{k-1} is known. Furthermore, we assume that the voltages at time points t_k and t_{k-1} are calculated. Between the two time points, the voltage is modeled as a perfect ramp, which is the general assumption in LTE estimation. We will consider the LTE estimation for a first-order admittance

$$Y(s) = \frac{k}{s-p} \tag{7.81}$$

Then the exact solution of the state variable x at time point t_k is

$$\tilde{x}(t_k) = e^{p\Delta t_k} x(t_{k-1}) + \frac{e^{p\Delta t_k}(p\Delta t_k - 1) + 1}{\Delta t_k \cdot p^2} v(t_{k-1}) + \frac{1}{\Delta t_k}\left(-\frac{1}{p^2} - \frac{1}{p}\Delta t_k + \frac{1}{p^2} e^{p\Delta t_k}\right) v(t_k) \tag{7.82}$$

It is clear that using recursive convolution with a piece-wise linear waveshape model yields the solution in (7.82), hence there is no error.

At each time point:

- Calculate intermediate variables, store them in vector **x**:

$$\begin{bmatrix} x_i^{(11)} \\ \vdots \\ x_i^{(21)} \\ \vdots \\ x_i^{(12)} \\ \vdots \\ x_i^{(2)} \end{bmatrix} \leftarrow \begin{bmatrix} \zeta_i^{(11)} \cdot x_i^{(11)} + \eta_i^{(11)} \cdot v_1(t_{k-1}) \\ \vdots \\ \zeta_i^{(21)} \cdot x_i^{(21)} + \eta_i^{(21)} \cdot v_1(t_{k-1}) \\ \vdots \\ \zeta_i^{(12)} \cdot x_i^{(12)} + \eta_i^{(12)} \cdot v_2(t_{k-1}) \\ \vdots \\ \zeta_i^{(22)} \cdot x_i^{(22)} + \eta_i^{(22)} \cdot v_2(t_{k-1}) \end{bmatrix}$$

- Calculate the equivalent current sources:

$$\begin{bmatrix} I_{1eq} \\ \\ I_{2eq} \end{bmatrix} \leftarrow \begin{bmatrix} \sum_{i=1}^{q} k_{11i} \cdot x_i^{(11)} + \sum_{i=1}^{q} k_{12i} \cdot x_i^{(12)} \\ \sum_{i=1}^{q} k_{21i} \cdot x_i^{(21)} + \sum_{i=1}^{q} k_{22i} \cdot x_i^{(22)} \end{bmatrix}$$

- Stamp equivalent conductances and current sources into MNA matrix and solve for the voltages
- Update the state vector for the next time point.

$$\begin{bmatrix} x_i^{(11)} \\ \vdots \\ x_i^{(21)} \\ \vdots \\ x_i^{(12)} \\ \vdots \\ x_i^{(2)} \end{bmatrix} \leftarrow \begin{bmatrix} x_i^{(11)} + \theta_i^{(11)} \cdot v_1(t_k) \\ \vdots \\ x_i^{(21)} + \theta_i^{(21)} \cdot v_1(t_k) \\ \vdots \\ x_i^{(12)} + \theta_i^{(12)} \cdot v_2(t_k) \\ \vdots \\ x_i^{(22)} + \theta_i^{(22)} \cdot v_2(t_k) \end{bmatrix}$$

FIGURE 7.9 Flowchart of equivalent current source updating scheme of a two-port.

7.6 Summary

We have described two methods to include interconnect macromodels into SPICE-like time-domain nonlinear circuit simulators. The first one relies on finding a state-space realization of the reduced order model. We have shown that once such a realization is obtained, it can be synthesized with an equivalent circuit so that any circuit simulator can be used. The second method, based on the recursive convolution, requires a modification in the simulators. In this approach the multiports are replaced by their discrete time-domain companion models. The parameters of these companion models are calculated with a constant cost at each time step by exploiting the fact that the pole-residue descriptions of the macromodels are available.

References

[7.1] G. H. Golub and C. F. Van Loan, *Matrix Computations,* Third Edition. Baltimore: The Johns Hopkins University Press, 1996.

[7.2] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. DuCroz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen, *LAPACK Users' Guide, Release 2.0,* Second Edition. Philadelphia, 1995.

[7.3] C. -T. Chen, *Linear System Theory and Design,* Third Edition. New York: Oxford University Press, 1999.

[7.4] J. R. Griffith and M. S. Nakhla, "The-domain analysis of lossy coupled transmission lines," *IEEE Transactions on Microwave Theory and Techniques,* vol. MTT-38, pp. 1480-1487, Oct. 1990.

[7.5] J. E. Schutt-Aine and R. Mittra, "Scattering parameter transient analysis of transmission lines loaded with nonlinear terminations," *IEEE Transactions on Microwave Theory and Techniques,* vol. MTT-36, pp. 529-536, 1988.

[7.6] A. Semlyen and A. Dabuleanu, "Fast and accurate switching transient calculations on transmission lines with ground return using recursive convolutions," *IEEE Trans. Power Apparatus and Systems,* vol. PAS-94, no.2, 1975.

[7.7] V. Raghavan, E. Bracken, and R. A. Rohrer, "AWESpice: A general tool for the accurate and efficient simulation of interconnect problems," in *Proc. 29th ACM/ IEEE Design Automation Conf.,* pp. 87-92,1992.

[7.8] F. -Y. Chang, "Transient simulation of nonuniform coupled lossy transmission lines characterized with frequency-dependent parameters, Part II: discrete time analysis," *IEEE Trans. Circuits and Systems - I: Fundamental Theory and App.,* vol. 39, no. 11, Nov. 1992.

[7.9] S. Lin and E. S. Kuh, "Transient simulation of lossy interconnects based on the recursive convolution formulation," *IEEE Trans. Circuits and Systems - I: Fundamental Theory and App.,* vol. 39, no. 11, Nov. 1992.

[7.10] Y. Liu, L. T. Pileggi, and A. J. Strojwas, "ftd: Frequency to time domain conversion for reduced-order interconnect simulation," *IEEE Trans. Circuits and Systems - I: Fundamental Theory and App.,* no. 4. Apr. 2001.

[7.11] J. Vlach and K. Singhal, *Computer Methods for Circuit Analysis and Design,* Second Edition. New York: Van Nostrand Reinhold, 1994.

CHAPTER 8

Interfacing Interconnect and Gate-Delay Models

To shorten design cycles, digital systems are often designed at the gate and/or cell level. In contrast to designing at the transistor level, gate or cell level design can significantly reduce costly design verification by precharacterizing the gate and cell delays for static timing analysis. The cell delays and transitions are generally expressed empirically as a function of load capacitance and input signal transition time. However, with the emerging interconnect dominance, gate loads can no longer be modeled by purely capacitive loads for high performance digital circuits. In this chapter we propose methods for interfacing empirical gate models to R(L)C interconnect models. This is followed by application of the same modified gate models to capture dominant interconnect coupling effects.

8.1 Logic Stage Delay Calculation

For digital IC technologies in general, and CMOS in particular, it is assumed that the load at the output of the gate has a negligible impact on the waveform behavior at the gate input. Therefore, digital circuit delay calculation along timing paths can be performed gate by gate or logic stage by logic stage for applications such as timing analysis [8.1], as shown in Figure 8.1. It should be noted that the gate-source and gate-drain capacitances are included as part of the gate characterization and, therefore, not considered during the partitioning process.

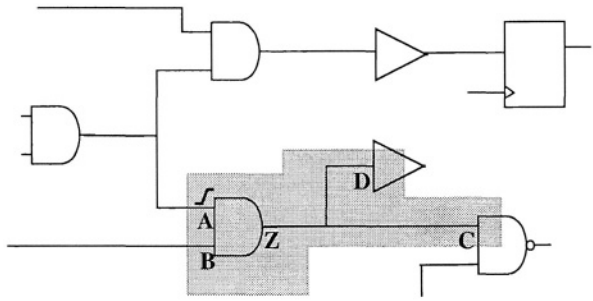


FIGURE 8.1 Definition of a logic stage (shown in the shaded area).

Delay calculation is performed for each logic gate/stage input individually. It is assumed that all other gate inputs are set to logic values such that the input pin under consideration can cause the gate output to transition. The outputs that are expected from a delay calculator are the waveforms at the fanouts, as a function of the waveform at the switching input pin. For example when we look at the logic stage in Figure 8.1, the waveforms at Z, D and C pins are computed in the delay calculator for when A pin is switching from low to high. Notice that a separate computation is required when A switches from high to low or when B switches.

8.2 Gate Characterization

There are two approaches to gate delay modeling which have gained widespread acceptance: 1) empirically derived expressions or look-up tables for delay and output-signal transition as a function of load capacitance and input-signal transition time (*k*-factor equations) [8.2]; and 2) a switch-resistor model comprised of a linear resistor and a step function of voltage [8.3]. Both methods are empirically-based, since even the second method requires empirical fitting to approximate the resistance value [8.5].

Switch resistor models have the advantage that their coupling with the RC interconnect is inherently modeled. That is, the resistance model is able to capture the interaction of the gate's output resistance and the RC load. Timing analysis tools such as TV [8.4] and Crystal [8.5] were developed using switch-resistor models to analyze the transistor level circuit descriptions. The main difficulty with these approaches is calculating a single linear resistor which captures the switching behavior of a CMOS gate. Recognizing that this resistance is a function of the gate's input signal transition

time and output load, in [8.5] a single output resistance for the gate is empirically derived. That is, the resistance is calculated as the average output impedance over a range of input signal transition times and output loads.

When the load is purely capacitive, one can completely precharacterize a gate's delay and output signal behavior as a function of input signal transition time, $t_{\text{tr-in}}$, and load capacitance, C_L [8.2]. The experimental data for the delay, t_d, and the gate-output waveform transition time, $t_{\text{tr-out}}$, are generally fitted to *k*-factor equations:

$$t_d = k(t_{\text{tr-in}}, C_L) \tag{8.1}$$

$$t_{\text{tr-out}} = k'(t_{\text{tr-in}}, C_L) \tag{8.2}$$

where k and k' are empirically fitted functions, and the delay and transition times are defined as shown in Figure 8.2. Note that we have simplified the waveforms as saturated ramps by fitting a line through two predefined characterization points. With this approximation it is possible to represent the waveshape with a single number, the transition time. Sometimes it is useful to keep the rail-to-rail value instead of remembering the characterization points. The extended transition time, t_{ex}, can be computed as

$$\begin{aligned} t_{ab} &= |t_b - t_a| \\ t_{ex,\,ab} &= t_{ab}/|a - b| \\ 0 < a, b < 1 \end{aligned} \tag{8.3}$$

where t_k denotes the time at which the waveform is at k percent of its final value.

Some methodologies use more than two points to characterize the waveshape, while others employ more complex load models -- both of which increase the complexity of delay calculation significantly. These types of characterizations are mainly used for back-end verification processes for which speed is not critically important. For notational simplicity in this chapter, we assume the waveshape to be a saturated ramp. The generalization beyond simple ramps will be discussed as required.

The delay and output transition time can also be characterized via look-up tables. The construction of such a table is illustrated in Figure 8.3. Throughout this chapter, we assume that gate delays are characterized with look-up tables. But, the methods we present are equally applicable to *k*-factor delay models (regression fit of table data) as well.

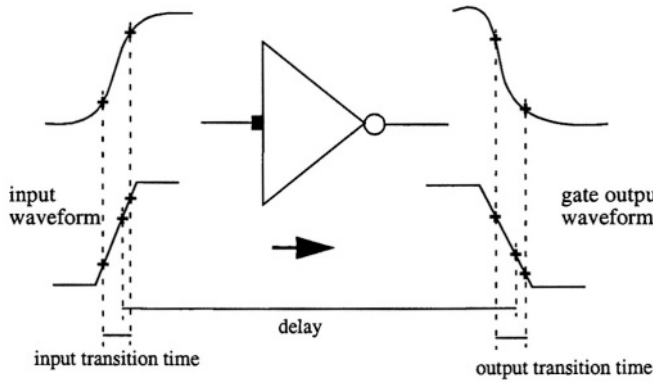
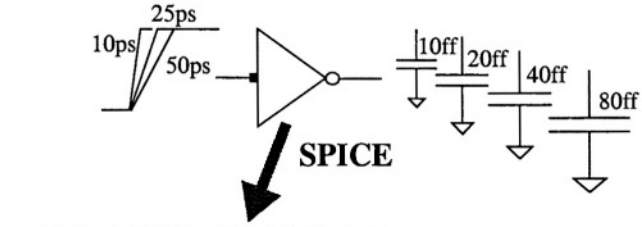


FIGURE 8.2 Waveshape approximation.



Output falling delay table (ps)

output load (ff)

input transition (ps)	10	20	40	80
10	47	73	110	143
25	50	77	116	148
50	51	81	122	151

Indicates that when load is 80 ff and input is 50 ps, gate delay is 151 ps.

FIGURE 8.3 Emprical delay characterization of a gate with a look-up table.

8.3 Effective Capacitance Concept

Due to the increase in total metal resistance with scaling, and the tendency for the *effective* gate output resistances to decrease as technologies are advanced, the RC shielding effect becomes significant for deep submicron CMOS. To illustrate this point, consider a simple gate model driving a distributed RC interconnect, with a load capacitance at the end of the line, as shown in Figure 8.4. Assume that the gate is modeled by a resistance and a Thevenin voltage source that are functions of input transition time and output capacitance load. In this case, assume that the gate output resistance, R_d, and Thevenin voltage signal were selected as those values that would yield the same output delay and transition time as the actual gate when the load is the total capacitance, $C_M + C_L$.

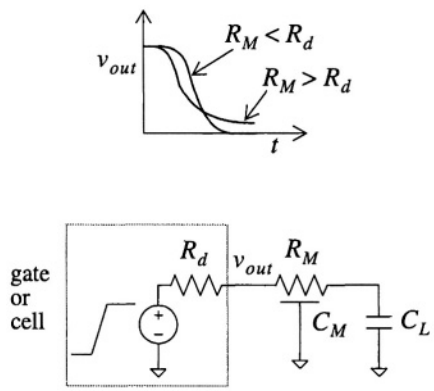


FIGURE 8.4 Gate driving RC load for different metal resistance.

If $R_d \gg R_M$, then the gate delay is accurately characterized by the empirical model as a function of total capacitance. However, if we consider the same gate resistance and total load capacitance, but increase the metal resistance so that $R_d \ll R_M$, then the gate delay at node v_{out} will decrease. This decrease in delay is due to the metal resistance shielding a portion of the downstream load capacitance. The difference in responses are sketched in Figure 8.4. Note that the gate delay decreases, but the overall delay at C_L would increase due to an increase in R_M. We should also point out that the responses for lines with significant metal resistance also tend to have non-digital shaped waveforms as shown.

In order to preserve the simplicity and efficiency of the empirical gate models for complex RC loads, one can map the complex load to an *effective capacitance (Ceff)* [8.6]. Since its invention, the Ceff approach has been successfully used in the design of high-speed ICs. For an example, refer to [8.7], where the importance of modeling the effective capacitance loading was analyzed for an industrial microprocessor design.

8.4 Two-Step Delay Approximation

The simplest model of a gate and corresponding interconnect delay would be a two-step approximation, as shown in Figure 8.5. The complete stage delay is the sum of the gate and interconnect delays. To capture the gate delay using a simple empirical gate delay model, the RC load is replaced by the effective capacitance, C_{eff}, and then the gate delay and output transition time is obtained from the empirical model of the gate. Once the transition time at the gate output is calculated, the gate output waveform is approximated with a saturated ramp. The interconnect delay is then calculated using this saturated ramp waveform as the input excitation.

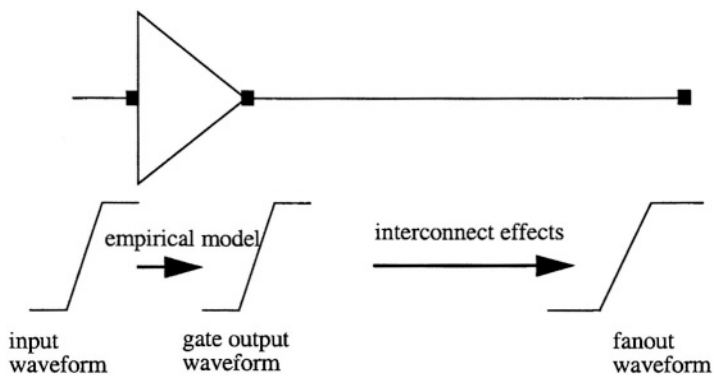


FIGURE 8.5 Two-step delay calculation.

The two-step delay approximation works well when the load seen by the gate is accurately approximated by the total capacitance of the net. That is, if the metal resistance is negligible, the whole interconnect behaves like an equipotential surface and the waveform at the gate output appears instantly at all fanouts -- i.e., there is no slope degradation or delay from gate output to fanouts.

8.5 Thevenin Delay Modeling

With the increasing effects of interconnect resistance, however, gate output waveforms become increasingly non-digital and can no longer be modeled as saturated ramps. A solution to this problem is to use Thevenin gate models based on the Ceff concept, as proposed in [8.8] [8.9]. The gate is replaced by a time-varying voltage source and a constant resistor, as shown in Figure 8.6. Based on a gate input waveform, A, the gate is modeled as a linear Thevenin equivalent with a time-varying voltage waveform, T. The voltage waveshape T is iteratively determined as a function of the effective capacitance. Once the Thevenin model is parameterized, the interconnect is then attached to the linear gate model and a linear circuit evaluation is performed. With this modeling approach, the RC nature of the interconnect can be captured to obtain the gate output and fanout waveforms more accurately.

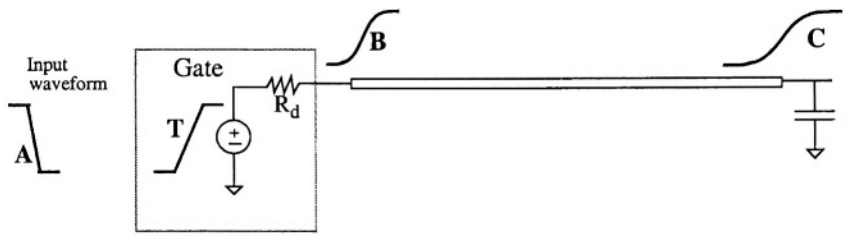
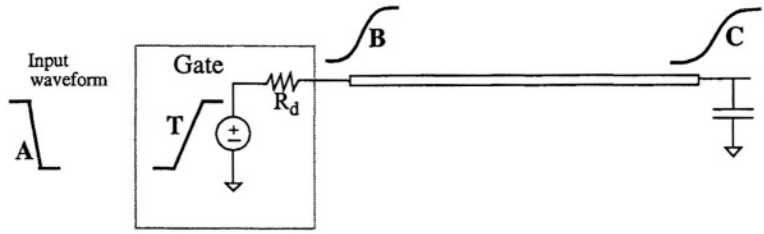


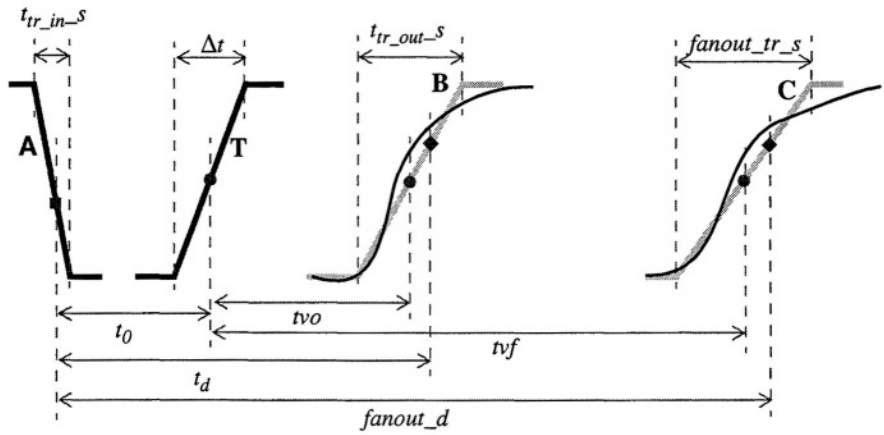
FIGURE 8.6 Thevenin gate model.

8.5.1 Construction of the Thevenin Model

The Thevenin voltage waveform T is generally modeled via a saturated ramp voltage that is characterized by a transition time Δt and a delay t_0. These parameters, along with some other delay and slew rate definitions, are illustrated in Figure 8.7.

The values of the Thevenin model parameters, t_0, Δt, and R_d, are chosen such that the waveshape at the model output and the actual waveform from the library lookup match. However, since the look-up tables and *k*-factor equations are not defined for noncapacitive loads we need to use an intermediate effective capacitance value, C_{eff}. To find C_{eff}, a commonly used procedure is to compute a capacitance value such that when the driver is the Thevenin model, the average current into the interconnect and C_{eff} are equal. Once C_{eff} is computed, it is used in the library lookup to obtain the gate output (delay and transition time). Then a Thevenin voltage is computed to match the library response.





- ■ input switching thresh
- ● standard 50% point
- ◆ output switching thresh

<*val*>_*s* = <*val*> * (upper_slew_thresh - lower_slew_thresh) / slew_derate;

t_{tr_in} := transition value at the gate input pin
t_d := delay value obtained from library when load is C_{eff}
t_{tr_out} := transition value obtained from library when load is C_{eff}

FIGURE 8.7 Definitions of variables used in Thevenin gate model computation.

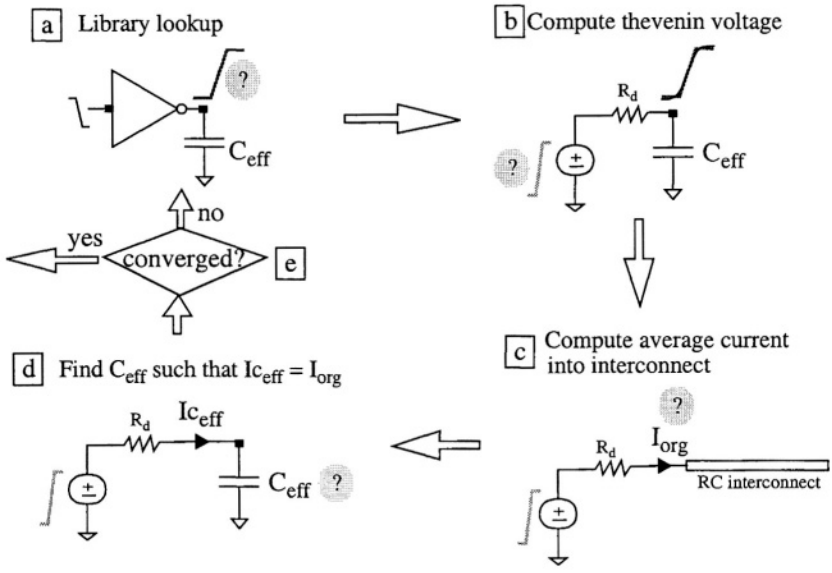


FIGURE 8.8 Thevenin gate model iterations. At each step we solve for (?).

The process of Thevenin delay modeling is illustrated in Figure 8.8. Mathematically, it is equivalent to solving the following set of nonlinear equations:.

$$\begin{cases} t_d = k(t_{\text{tr-in}}, C_{\text{eff}}) \\ t_{\text{tr-out}} = k'(t_{\text{tr-in}}, C_{\text{eff}}) \end{cases}$$

$$\begin{cases} t_0 = f(t_d, t_{\text{tr-out}}, C_{\text{eff}}, R_d) \\ \Delta t = f'(t_{\text{tr-out}}, C_{\text{eff}}, R_d) \end{cases} \tag{8.4}$$

$$\{I_{\text{avg}}(C_{\text{eff}}, \Delta t, t_0, R_d) = I_{\text{avg}}(\pi, \Delta t, t_0, R_d)$$

where t_0 and Δt are the parameters we wish to obtain. The first two equations in (8.6) are the *k*-factor equations, and in the case of a table characterization they represent the table look-up process. The third and fourth equations correspond to the con-

struction of the Thevenin model. The last equation matches the currents in the actual and Ceff circuit.

We represent the actual interconnect circuit with a pi symbol since we use a reduced order pi-model to approximate the driving-point characteristic of the RC load. The pi-model has been observed to provide sufficient accuracy for the purpose of driving point modeling inRC circuits. What is not shown in (8.4) is the selection of the Thevenin resistance R_d, which is addressed in Section 8.6.3.

We solve (8.4) iteratively to find t_0 and Δt. As shown in Figure 8.8, we can divide each iteration stage into five steps. In the following we explain these steps in detail and describe their correspondence to (8.4).

Precomputations

Certain parameters such as the moments of the interconnect, driving point load model, and gate resistance R_d can be calculated before the iteration process. Also prior to iterating we can determine if we need to perform a full Ceff model delay calculation or a simpler model will suffice. Refer to sections 8.6.1 and 8.6.2 for a detailed treatment of the subject. For the computation of the gate resistance, refer to section 8.6.3.

Iterations

a) *Library lookup:* From the empirical gate delay tables, obtain the gate output delay, t_d, and the gate output transition time, $t_{\text{tr-out}}$, for the given C_{eff} and input transition time, $t_{\text{tr-in}}$, values. The initial guess for C_{eff} in the first iteration is C_{total}, the total capacitance of the interconnect. This step corresponds to evaluating the first two equations.

b) *Compute Thevenin voltage:* If necessary, first update the R_d value (Section 8.6.3). Then find the Thevenin voltage parameters, t_0 and Δt, such that the output of the R_d-C_{eff} circuit in Figure 8.8(b) matches the waveform obtained from the table in the library lookup step. This step is represented in (8.4) by the functions f(.) and f'(.) and on the details of derivations and computations for obtaining the Δt and t_0 values are outlined in Section 8.6.4.

c) *Find* $I_{avg}(\pi)$*:* Given t_0, Δt, R_d, and the load model (Section 8.6.1) for the interconnect, compute the average current into the interconnect, $I_{avg}(\pi)$. The details are given in Section 8.6.5.

d) *Compute* C_{eff}*:* We find C_{eff} such that

$$I_{avg}(C_{eff}) = I_{avg}(\pi) \tag{8.5}$$

where $I_{avg}(C_{eff})$ is the average current into the circuit in Figure 8.8(d). We later show that it is given by

$$I_{avg}(C_{eff}) = \frac{C_{eff}}{\Delta t} + \frac{R_d C_{eff}^2 e^{\frac{-\Delta t}{R_d C_{eff}}}}{\Delta t^2} - \frac{R_d C_{eff}}{\Delta t^2}. \tag{8.6}$$

Defining $x = R_d C_{eff}$ and multiplying each side by $R_d \Delta t^2$ yields

$$R_d \Delta t^2 \cdot I_{avg}(\pi) = x\Delta t + x^2\left(e^{\frac{-\Delta t}{x}} - 1\right). \tag{8.7}$$

The right side of equation (8.7) is a smooth function of x and bounded by $\Delta t^2/2$. A regula-falsi scheme can be employed to iteratively solve for x in (8.7).

e) *Converged?:* The steps are repeated until the values of t_0 and Δt converge. Alternatively, we can check for the convergence of C_{eff}. Typically it takes 2-3 iterations to converge on the C_{eff} value. However, there are certain cases where C_{eff} is only 10% of C_{total} (for example, a long interconnect with a large load at the end). In these cases, a larger number of iterations may be required.

After the iterations

To get the waveforms at the gate output or the fanouts we need to simulate the interconnect and the Thevenin gate model together. The problem at hand is a very familiar one: linear circuit analysis with a saturated ramp voltage source driver. For each fanout of the net, we need to compute *tvf* and *fanout_tr_s* as defined in Figure 8.7 (*tvf* is the 50%-50% delay between the saturated ramp and the fanout waveform; *fanout_tr_s* is the rail to rail transition time of the fanout waveform). For details, refer

to Section 8.6.6. When *tvf* and *fanout_tr_s* are found we can convert them to *fanout_d and fanout_tr* using:

$$fanout\_d = t_0 + tvf + (output\_thresh - 0.5) * fanout\_tr\_s \quad (8.8)$$

$$fanout\_tr = fanout\_tr\_s \,/\, slew\_adjust * rc\_slew\_derate \quad (8.9)$$

8.6 Thevenin Delay Model Computation Details

In the following subsections we describe the details of the Thevenin gate model computation. Note that it is possible to employ entirely different strategies for each step. Here, some commonly used and simple approaches are explained for completion of the Ceff model description.

8.6.1 Computation of the Load Model

Rather than analyzing the entire interconnect at each iteration to calculate the average current flowing into it, we can find a reduced order driving point model during the precharacterization step and use the reduced model during the iterations. Although more general Cauer realizations [8.10] are possible, a simple pi-model (as shown in Figure 8.9), which is equivalent to a two pole realization of an RC driving point, is often accurate enough for most typical CMOS technologies and circuits. The pi-circuit has is popular for driving-point characteristic modeling since it is the simplest model which captures some resistive shielding [8.11].

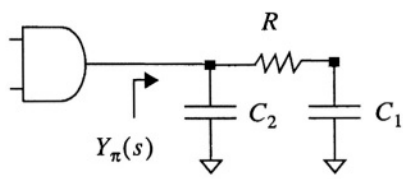


FIGURE 8.9 Pi-circuit for driving point modeling.

The pi-model can be easily obtained from the moments. Let $Y(s)$ be the driving point admittance function of the actual gate load and y_i be the i^{th} moment of $Y(s)$:

$$Y(s) = y_0 + y_1 s + y_2 s^2 + y_3 s^3 + \ldots \quad (8.10)$$

where y_0 is zero for RC trees. Earlier chapters have shown how these moments are calculated efficiently. The driving point admittance of the pi-circuit can be written directly

$$Y_\pi(s) = sC_2 + \frac{sC_1}{1 + sRC_1} \tag{8.11}$$

By matching (8.10) to (8.11) up to third order we obtain the parameters for the pi-circuit model which will match the first three moments of the actual load:

$$\begin{aligned} C_1 &= y_2^2/y_3 \\ C_2 &= y_1 - C_1 \\ R &= -y_2/C_1^2 \end{aligned} \tag{8.12}$$

8.6.2 Is Ceff Model Needed?

For interconnects with a small resistance value, C_{eff} can be very close to C_{total}. In these cases the interconnect behaves as an equipotential surface, i.e., there is no signal degradation because of the interconnect. To catch this situation prior to the start of iterations, we can compare the pi-model resistance to the Ceff gate model resistance.

Similarly, we can define a maximum frequency of interest as

$$f_{\max} = \frac{1}{t_r} \tag{8.13}$$

where t_r is the transition time at the gate output due to C_{total}. If at that frequency the impedance of C_1 is much larger than R, we can ignore R since it will have no effect at lower frequencies. Therefore, if the filtering criteria of

$$R \ll t_r/(2\pi C_1) \tag{8.14}$$

holds, we can simply use C_{total} to calculate the gate delay.

8.6.3 R_d - Gate Resistance

The gate resistance can be derived of varying complexity. It can be created as a function of input transition time as multiple resistance values, or as a single constant R_d. The benefits of a single R_d value are obvious [8.9], and the corresponding Ceff process is greatly simplified.

An R_d value can be precomputed [8.8] and the Ceff iterations can be started using this value. Although the Ceff method is fairly insensitive to the R_d value, we can change it adaptively during the iterations to further increase the accuracy. Experimentally it has been observed that gate resistance should be as large as possible. If a small R_d is selected, Δt will be slower and closer to gate output transition time, $t_{tr\text{-}out}$. This is not desired since it means that the Ceff gate output will look more like a saturated ramp, which we are trying to avoid. Therefore it makes sense to select as large R_d as possible to make the Ceff gate output waveform look more realistic. However, there is a limit on how large we can select R_d. We have to select it small enough so that the output of the R_d-C_{eff} circuit in Figure 8.8b can have a signal with a transition time of $t_{tr\text{-}out}$.

8.6.4 Computation of Thevenin Voltage Parameters

At step b in Figure 8.8 we find the Thevenin voltage parameters t_0 and Δt such that the output waveform of the R_d-C_{eff} circuit matches the waveform obtained from the empirical model. Recall that the empirical model output is represented by the delay (t_d) and transition time ($t_{tr\text{-}out}$) pair. Assuming a rising output transition, the (normalized) output of the R_d-C_{eff} circuit can be written as

$$v_o(t-t_0,\Delta t) = \begin{cases} 0 & \text{if } t-t_0<0 \\ \dfrac{v_{ramp}(t-t_0)}{\Delta t} & \text{if } 0\le t-t_0<\Delta t \\ \dfrac{v_{ramp}(t-t_0)}{\Delta t}-\dfrac{v_{ramp}(t-\Delta t-t_0)}{\Delta t} & \text{if } t-t_0\ge\Delta t \end{cases} \tag{8.15}$$

where $v_{ramp}(t)$ is the unit ramp response

$$v_{ramp}(t) = t - R_d C_{eff}\cdot\left(1-e^{\frac{-t}{R_d C_{eff}}}\right) \tag{8.16}$$

To find the Δt value we need to solve the following nonlinear equation systems:

$$\begin{aligned} v_o(t_x - t_0, \Delta t) &= x \\ v_o(t_y - t_0, \Delta t) &= y \\ \frac{t_y - t_x}{y - x} &= t_{\text{tr-out}} \end{aligned} \tag{8.17}$$

In the last equation of (8.17), we match the transition times of the empirical model and the R_d-C_{eff} circuit assuming a x-y (i.e. 20%-80%) slope characterization for the exponential waveform. Note that the Δt value is independent from t_0.

Instead of solving these nonlinear equations at each Ceff iteration, we can precharacterize the solution with 2-d look-up tables similar to *k*-factor representation:

$$\Delta t = f'(R_d \cdot C_{\text{eff}}, t_{\text{tr-out}}) \tag{8.18}$$

which is the fourth equation in (8.6). Note that for each different x%-y% slope characterization, a different Δt empirical model is required.

Once Δt is found we can calculate the value of tvo from (8.15). Therefore, a similar precharacterization for tvo (delay between gate output waveform and Thevenin waveform as defined in Figure 8.7) is possible:

$$\text{tvo} = f''(R_d \cdot C_{\text{eff}}, t_{\text{tr-out}}) \tag{8.19}$$

Finally, t_0 is obtained as

$$t_0 = t_d - \text{tvo} \tag{8.20}$$

which corresponds to the third equation in (8.6).

8.6.5 Computation of Average Current (I_{avg})

In this section we detail the steps for calculating the average currents for the two circuits shown in Figure 8.10. The average current is defined as the ratio of the total current flowing into the load model during the transition period of the input waveform to

the transition period. For example, if $i_\pi(t)$ is the driving point current waveform for the pi-model, the average current is given by

$$I_{\text{avg}}(\pi) = \frac{1}{\Delta t}\int_{t_0}^{t_0+\Delta t} i_\pi(t)dt \tag{8.21}$$

We next find a closed form expression for $I_{\text{avg}}(\pi)$, so that it can be easily evaluated during Ceff iterations. For this, we first write the input admittance function $Y(s)$ in terms of the circuit parameters:

$$Y(s) = \frac{s(s\alpha + \beta)}{s^2 a + sb + 1} \tag{8.22}$$

where

$$\begin{aligned} \alpha &= RC_1C_2 \\ \beta &= C_1 + C_2 \\ a &= R_dRC_1C_2 \\ b &= R_dC_2 + R_dC_1 + RC_1 \end{aligned} \tag{8.23}$$

Note that the input to the circuit is a saturated ramp waveform. However, we are interested in $i_\pi(t)$ only for the time interval from t_0 and Δt. Therefore, we consider $i_\pi(t)$ as the ramp response for the driving point current. In the s-domain it is given by

$$I_\pi(s) = V_{in}(s)Y(s) = \frac{1}{\Delta t \cdot s^2}Y(s) \tag{8.24}$$

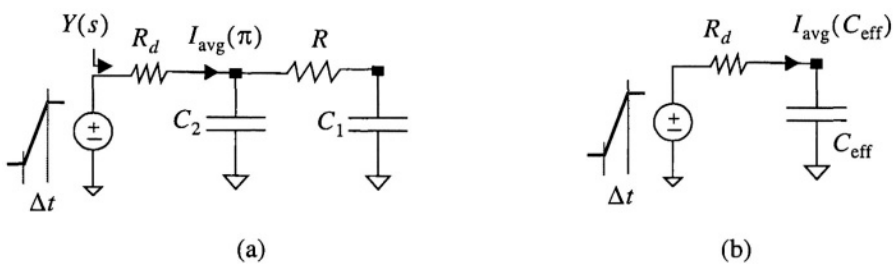


FIGURE 8.10 Average currents into (a) pi-circuit and (b) Ceff load.

From (8.22) and (8.24) it follows that

$$I_\pi(s) = \frac{1}{\Delta t \cdot s}\left(\frac{k_1}{s-p_1} + \frac{k_2}{s-p_2}\right) \tag{8.25}$$

where

$$\begin{aligned} p_1 &= \frac{-b+\sqrt{b^2-4a}}{2a} \qquad & p_2 &= \frac{-b-\sqrt{b^2-4a}}{2a} \\ k_1 &= \frac{p_1\alpha+\beta}{\alpha(p_1-p_2)} \qquad & k_1 &= \frac{p_2\alpha+\beta}{\alpha(p_2-p_1)} \end{aligned} \tag{8.26}$$

From (8.25), we obtain the current waveform as

$$i_\pi(t) = \frac{1}{\Delta t}\left(\frac{k_1}{p_1}e^{p_1\Delta t} + \frac{k_2}{p_2}e^{p_2\Delta t} - \frac{k_1}{p_1} - \frac{k_2}{p_2}\right) \qquad 0 \le t \le \Delta t \tag{8.27}$$

Inserting (8.27) into (8.21) and after some algebra, we obtain

$$I_{\text{avg}}(\pi) = \frac{1}{\Delta t^2}\left(\frac{k_1}{p_1^2}e^{p_1\Delta t} + \frac{k_2}{p_2^2}e^{p_2\Delta t} + (\alpha - \beta b) + \beta\Delta t\right) \tag{8.28}$$

In (8.28), the parameters k_1, k_2, p_1, p_2, and b are R_d dependent, thus they need to be updated whenever R_d changes. Note that (8.28), which is derived for a second order model, can be easily obtained for higher order load models also.

In a similar manner, the current waveform in the case of Ceff load is given by

$$i_{\text{Ceff}}(t) = \frac{C_{\text{eff}}}{\Delta t}\left(1 - e^{\frac{-t}{RC_{\text{eff}}}}\right) \qquad 0 \le t \le \Delta t \tag{8.29}$$

After evaluating the integration, we obtain

$$I_{\text{avg}}(C_{\text{eff}}) = \frac{C_{\text{eff}}}{\Delta t} + \frac{R_d C_{\text{eff}}^2 e^{\frac{-\Delta t}{R_d C_{\text{eff}}}}}{\Delta t^2} - \frac{R_d C_{\text{eff}}}{\Delta t^2} \tag{8.30}$$

8.6.6 Calculation of the Delay at the Fanouts

Once the Thevenin model is obtained for the gate -- based on convergence to a C_{eff} model (Δt, t_0, and R_d) -- we can solve the equivalent circuit shown in Figure 8.11 to find the delays at the fanout nodes. In general, such a linear circuit can be analyzed using one of the many delay calculation methods explained in the previous chapters of this book.

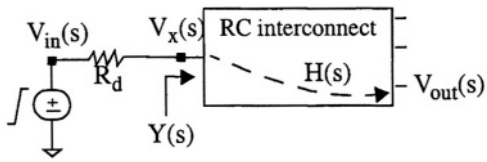


FIGURE 8.11 Circuit to compute fanout waveform.

These methods usually have two steps: (1) moment calculation from the circuit and (2) delay computation from the moments. In this case, the first step can be skipped since we already know the moments of the input admittance and transfer functions of the interconnect circuit (at least to find the pi model). Referring to Figure 8.11, we have the relation

$$H_1(s) = \frac{H(s)}{1 + R_d Y(s)} \tag{8.31}$$

where $H_1(s)$ is defined as

$$V_{\text{out}}(s) = H_1(s) V_{\text{in}}(s) \tag{8.32}$$

Given the moments of $Y(s)$ and $H(s)$,

$$\begin{aligned} Y(s) &= y_0 + y_1 s + y_2 s^2 + y_3 s^3 + \ldots \\ H(s) &= h_0 + h_1 s + h_2 s^2 + h_3 s^3 + \ldots \end{aligned} \tag{8.33}$$

we wish to compute the moments of $H_1(s)$,

$$H_1(s) = m_0 + m_1 s + m_2 s^2 + m_3 s^3 + \ldots \tag{8.34}$$

For this we first substitute the moment expansions of $Y(s)$ and $H(s)$ in (8.33) into (8.31), and then use (8.34) to obtain

$$\frac{h_0 + h_1 s + h_2 s^2 + \ldots}{1 + R_d(y_0 + y_1 s + y_2 s^2 + \ldots)} = m_0 + m_1 s + m_2 s^2 + \ldots \tag{8.35}$$

After cross-multiplying and then matching the s terms, we obtain

$$\begin{aligned} m_0 &= h_0 \\ m_1 &= h_1 - R_d y_1 \\ m_2 &= h_2 - R_d y_2 - m_1 R_d y_1 \\ m_3 &= h_3 - R_d y_3 - m_1 R_d y_2 - m_2 R_d y_1 \end{aligned} \tag{8.36}$$

Once the moments of $H_1(s)$ and Δt are known, depending on the desired accuracy, any delay estimation or calculation method described earlier in this book can be used to obtain the delay value at the fanout.

8.7 Gate Models for General RLC Loading

As interconnect inductance effects become more pronounced for long, wide wires, and as supply voltages scale down, shapes at the gate outputs become more and more nondigital and the single saturated ramp may not be enough to model the Thevenin voltage source. The Thevenin model accuracy can be increased by using a more general piecewise linear model [8.12], as shown in Figure 8.12.

In this model, the total transition time is split into pieces such that

$$v_{\text{th}}(t_0 + \sum_{i=1}^{k} \Delta t_i) = \alpha_k \tag{8.37}$$

where α_k's are the predefined percentage points. It is natural to select these points to be equally spaced in order to minimize the number of parameters required to characterize the waveform. Thus, for an n-piece approximation

$$\alpha_k = \frac{k}{n} \tag{8.38}$$

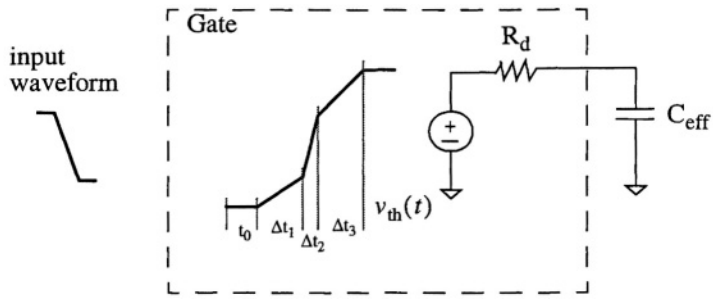


FIGURE 8.12 The piecewise linear Thevenin voltage source model.

Note that $n = 1$ reduces to the saturated ramp model. We construct the piecewise linear gate delay model, which is equivalent to finding the values of t_0 and Δt_i's, similar to the saturated ramp model except with small modifications in the steps corresponding to subfigures (b), (c), and (d) in Figure 8.8. We next explain the details.

Calculation of Thevenin Model

The Thevenin voltage source can be expressed in terms of unit ramp functions, $r(t)$, as

$$v_{th}(t) = \sum_{k=1}^{n} A_k r(t - t_k) \tag{8.39}$$

where $t_k = t_0 + \sum_{i=1}^{k} \Delta t_i$ and A_k's are the slopes which are functions of Δt_k's and α_k's. The waveform at the Ceff load becomes

$$v_{Ceff}(t) = \sum_{k=1}^{n} A_k [r(t - t_k) + R_d C_{eff}(e^{-(t-t_k)/(R_d C_{eff})} - 1)u(t - t_k)] \tag{8.40}$$

The problem is now, given a C_{eff} value, find Δt_k's. For this, we match the Ceff waveform to empirical model output at n different delay points

$$v_{Ceff}(t_{\beta_i}) = \beta_i \qquad 1 \le i \le n \tag{8.41}$$

where t_{β_i}'s are obtained from the *k*-factor equations or tables of the empirical model. This requires the characterization of the models for at least *n* delay points:

$$t_{\beta_i} = k_{\beta_i}(t_{\text{tr-in}}, C_L) \tag{8.42}$$

Calculation of Ceff

Similar to the RC interconnect loading case, we find C_{eff} by matching the currents flowing into the Ceff load and actual load. To calculate the current flowing into the actual load, previously we have modeled the driving point characteristics with a pi-circuit. But a more accurate model for general RLC loads, including the effect of the Thevenin resistance R_d, can be found using AWE. For example, a *q*-pole AWE characterization is given in the form

$$Y_{\text{load}}(s) = \sum_{i=1}^{q} \frac{k_i}{s - p_i} \tag{8.43}$$

This allows us to write the ramp response for the driving point current as

$$i_{\text{ramp}}(t) = \sum_{i=1}^{q} \frac{k_i}{p_i^2}(e^{p_i t} - 1)u(t) \tag{8.44}$$

where we have utilized the fact that $Y_{\text{load}}(0) = -\sum_{i=1}^{q} k_i/p_i = 0$ for RLC loads with no dc paths to ground. From (8.39) and (8.44), the current to the actual load is found as

$$i_{\text{load}}(t) = \sum_{k=1}^{n} A_k \cdot i_{\text{ramp}}(t - t_k) \tag{8.45}$$

The average current during the active interval, i.e., until V_{th} reaches its final value, can be obtained by evaluating the integral in (8.21) from t_0 to t_n:

$$\begin{aligned} I_{\text{avg}}(\text{load}) &= \frac{1}{t_n - t_0}\int_{t_0}^{t_n} i_{\text{load}}(t)dt \\ &= \frac{1}{t_n - t_0}\sum_{k=1}^{n} A_k\left(\sum_{i=1}^{q} \frac{k_i}{p_i^2}\left(t_k - t_n + \frac{1}{p_i}(e^{p_i(t_n - t_k)} - 1)\right)\right) \end{aligned} \tag{8.46}$$

The current waveform in the R_d-C_{eff} circuit is given by

$$i_{\text{Ceff}}(t) = C_{\text{eff}} \sum_{k=1}^{n} A_i (1 - e^{-(t-t_k)/(R_d C_{\text{eff}})}) u(t - t_k) \tag{8.47}$$

and the average current during the active interval becomes

$$I_{\text{avg}}(C_{\text{eff}}) = \frac{C_{\text{eff}}}{t_n - t_0} \sum_{k=1}^{n} A_i (t_n - t_k + R_d C_{\text{eff}} (e^{-(t_n - t_k)/(R_d C_{\text{eff}})} - 1)) \tag{8.48}$$

To find C_{eff}, we solve the nonlinear equation

$$I_{\text{avg}}(C_{\text{eff}}) = I_{\text{avg}}(\text{load}) \tag{8.49}$$

8.8 Interconnect Coupling

As described in Chapter 1, as IC dimensions scale, the multi-level interconnects are constructed such that the coupling capacitance becomes the dominant component of load capacitance. This effect is largely the result of the increased ratio between the lateral and the vertical capacitance of the line. This dominant coupling presents a difficult problem for gate and cell models that are characterized as a function of grounded output load capacitance. In addition, if the interconnect lengths and signal frequencies become such that inductance effects become evident, then modeling of the magnetic couplings might be required as well.

8.8.1 Coupled Interconnect Gate-Delay Calculation

The familiar delay computation becomes simulation of coupled RLC linear networks driven by CMOS gates, as shown in Figure 8.13. This section deals with the problem of combining coupled RLC networks with precharacterized gates when no transistor level information is present. We typically have two separate problems: One is the solution of the coupled interconnect waveforms when the input waveforms are given. The other problem is finding the worst/best case waveforms when only switching windows from timing analysis are known -- i.e. the exact waveform positions are unknown.

We first examine the traditional "Miller factor" approach to the coupling problem. Then in Section 8.8.3 more accurate delay/noise analysis is presented with the use of gate models for the coupled system. Lastly, Section 8.8.4 investigates how to compute worst/best case delays due to coupling on a victim net by "noise pulse method" when switching windows are available.

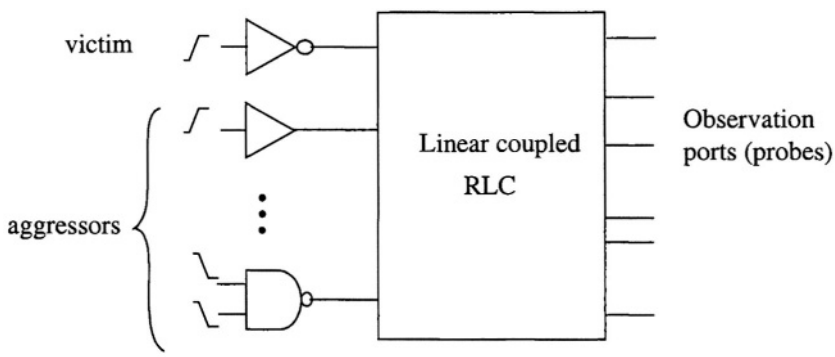


FIGURE 8.13 General gate level simulation problem with coupled interconnect.

8.8.2 Miller Capacitance Approach

A simple approach to approximating the impact of capacitive coupling is to use a factored "Miller capacitance" to ground in the place of the actual coupling capacitance. As shown in Figure 8.14, the coupling capacitance is replaced by a capacitance to ground, whose value is multiplied by a Miller factor[1]. Typically, the Miller factor is chosen as zero for min timing analysis and 2.0 for max timing analysis. This is an accurate approximation if the aggressor and victim lines and are perfectly symmetrical and switched at exactly the same time. But in general, for most problems other than busses, this is rarely the case. In addition, it should be noted that the zero and 2.0 Miller scaled models are *not* guaranteed worst case bounds [8.13]. In more elaborate schemes, the Miller factor can be a function of the aggressor's transition times and the switching windows. The functions are determined by experimentation. Despite all its inadequacies, this model is often used due to its simplicity.

1. Although this is not the true "Miller effect" that is well known for analog circuits [8.14], its similarity to that phenomena has resulted in it being used to denote these scaled coupling capacitances.

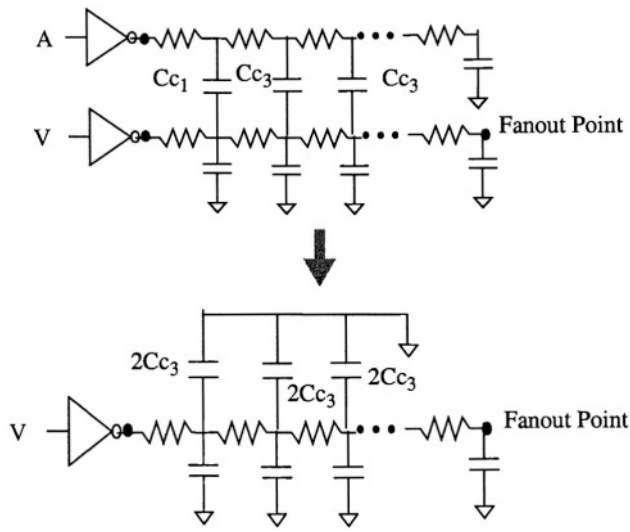


FIGURE 8.14 Miller capacitance modeling for victim net for max timing analysis.

8.8.3 Gate Models With Coupled R(L)C interconnect

As described in earlier sections of this chapter, we replace all of the gates with their Ceff gate models and iteratively solve for the gate switching behavior in terms of a Thevenin model. With coupling, the iterative procedure is very similar [8.15]. The basic Ceff principle is maintained: The average charge delivered by the Thevenin voltage source should be the same for the actual interconnect loading and the C_{eff} loading (Figure 8.8 c&d). The Thevenin equivalent parameters allow the model to fit two points of the actual gate response for the case of a capacitive load (Figure 8.8 steps a and b).

The only difference between the coupled system and the single gate system is in how to compute the average current for the actual loading (Figure 8.8 step c). For the single gate case, we solved for the average current as described in Section 8.6.5. For the coupled case, we solve for the linear system given in Figure 8.15, whereby we compute I_k, that is the total average current that is flowing into port k using

$$I_k = \sum_{j=0}^{n} \pm I_{jk}. \tag{8.50}$$

I_{jk} is the average current flowing into port k when port j is on (and the rest of the sources are set to zero). The signs in (8.50) are determined such that I_k is maximum when we are computing the worst case and the minimum when we are computing the best case. By assuming infinite switching windows at this stage, the resulting Ceff gate models are guaranteed to be pessimistic.

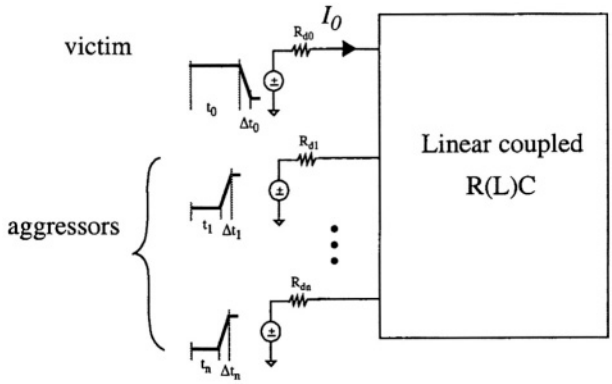


FIGURE 8.15 Circuit for computation of average current into actual interconnect.

After all I_k's are computed, steps d &a in Figure 8.8 are completed for each gate separately (I_{org} in Figure 8.8 will be replaced by I_k). The iterations are repeated until convergence in Ceff gate model parameters have been reached. Upon completion of the iterations, a multiport linear circuit analysis is performed to solve for the waveforms at the probe points. More on the details for computing the average current and noise waveforms in Section 8.8.5.

8.8.4 Computing worst and best case waveforms

Thus far we have considered the coupled interconnect problem when the positions of the input waveforms are known exactly. However, in a timing analysis environment, we only know the min-max ranges for the arrival and transition times at each gate input. The problem now becomes one of determining the worst-case alignment of aggressor signals as they impact the response on the victim.

To find the set of these variables that generates the best/worst case for the victim net is a difficult problem. Assuming that we start with some known switching points for the aggressors, upon Ceff iteration convergence we have a linear gate model for all

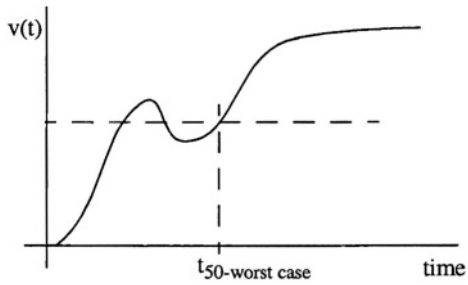


FIGURE 8.16 Composite victim waveform.

the aggressors and the victim. When combined with the exact interconnect, this system constitutes a large linear time-invariant system.

The Δt's of the Thevenin sources are already computed upon convergence of the Ceff iterations. From this information, the noise-free victim waveform and the noise bumps from each aggressor on the victim can be computed from the linear system. The problem of worst-case aggressor alignment now becomes one of finding the shifts in the Thevenin gate delay models, t_i's, which produce the slowest (fastest for min timing) victim waveform. Obviously the actual alignment problem is not linear, but the linear system produced within each Ceff iteration can be analyzed via superposition. It should be noted, however, that both the original signal and the noise waveforms are modified by their relative positions, which greatly complicates finding the worst case. Fortunately the observations made on the linear problem can be used to upper bound the worst case delay [8.15].

We investigate only the worst case scenario in detail here since the best case can be found by applying similar principles. In the worst case for capacitive coupling[1] we assume that the switching direction of the victim and the aggressors will be opposite. This fact can be found by observation. Since everything in the circuit is linear, we could simply add up the noise bumps and the noise-free victim waveform to find *composite victim waveform,* had we known the exact aggressor shifts. In such a waveform, the 50% switching point is defined as the last point the signal crosses 50% (see Figure 8.16). Due to crosstalk noise, the signal can cross the 50% several times, but the last crossing is the only value used for a static timing analysis.

It is possible to solve for the latest 50% crossing of the composite victim waveform using any root finding algorithm if the composite waveform is known exactly. But

1. Note that this can correspond to the best case for inductive coupling.

since the aggressor shifts are unknown, we still don't know the composite victim waveform. To find the worst-case alignment and the worst case 50% delay on the composite signal in a pessimistic way, we introduce the concepts of "noise pulse" and "noise combo-pulse." We begin for the case of a single aggressor.

Theorem 8.1: *We can approximate the "noise bump,"* $v_{aggressor}(t)$*, with an encapsulating pulse,* $v_{pulse}(t)$*, such that* $v_{pulse}(t) > v_{aggressor}(t)$ *for all t. This always results in a more pessimistic 50% delay of the composite victim waveform, regardless of the position of the pulse.*

Proof: Assume a falling victim waveform without loss of generality. Define $v_1(t) = v_{victim}(t) + v_{aggressor}(t)$ and $v_2(t) = v_1(t) + v_x(t)$ where $v_x(t) = v_{pulse}(t) + v_{aggressor}(t)$. It is important to observe that the last 50% crossing of $v_1(t)$ will always happen when it is falling. Since $v_x(t) > 0$ for all *t,* $v_2(t)$ will always cross 50% the last time later than $v_2(t)$. It follows that the result is pessimistic. QED

It may not be practical to use pulses that completely encapsulate the noise bump. For practical purposes, the pulse only covers $v_{aggressor}(t) > 0.1 v_{peak}$ (see Figure 8.17).

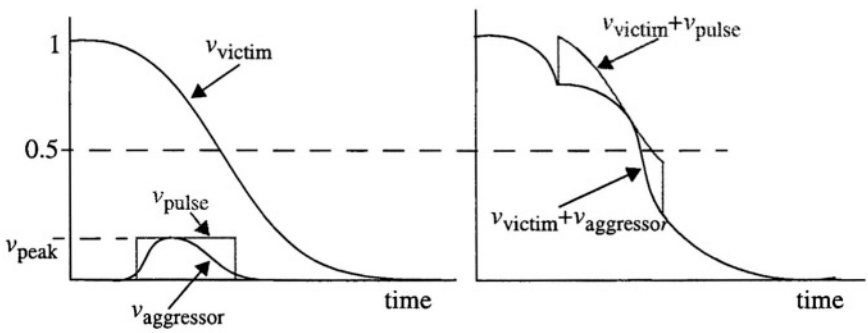


FIGURE 8.17 Noise pulse approximation to the noise bump.

Theorem 8.2: *Define the noise pulse to be*

$$v_{pulse}(t) = \begin{cases} v_{peak}, t_p < t < t_q \\ 0, \text{elsewhere} \end{cases} \quad (8.51)$$

Where v_{peak} is the peak of the noise bump. The values t_v and t_m are defined by:

$$\begin{aligned} v_{victim}(t_v) &= 0.5 \\ v_{victim}(t_m) &= 0.5 - v_{peak} \end{aligned} \tag{8.52}$$

where v_{victim} denotes the monotonic noise-free victim waveform. The worst case 50% delay, t_{50}, is always between t_v and t_m, independent of the position of the pulse. Moreover, if the pulse position (t_p and t_q) is known, it can be derived that

$$t_{50} = \begin{cases} \min(t_m, \max(t_q, t_v)), & t_p \le t_m \\ t_v, & t_p > t_m \end{cases} \tag{8.53}$$

Proof: Let $v_{cx}(t) = v_{victim}(t) + v_{peak}$. Observe from Figure 8.18 that $v_{cx}(t)$ crosses 50% at t_m. Let $v_c(t) = v_{victim}(t) + v_{pulse}(t)$. Since $v_{pulse}(t) \le v_{peak}$ for all t, $v_c(t) \le v_{cx}(t)$ for all t. Since $v_{cx}(t)$ is monotonic, 50% crossings of $v_c(t)$ will always be less than that of $v_{cx}(t)$. This proves that t_m is the maximum possible 50% delay for a given noise pulse with a peak voltage of v_{peak}, independent of the position of the pulse. Also observe that

$$v_c(t) = \begin{cases} v_{victim}(t) + v_{peak}, & t_p < t < t_q \\ v_{victim}(t), & \text{elsewhere} \end{cases}. \tag{8.54}$$

It is possible to draw $v_c(t)$ with different t_p and t_q's to come up with the formula in (8.53). QED

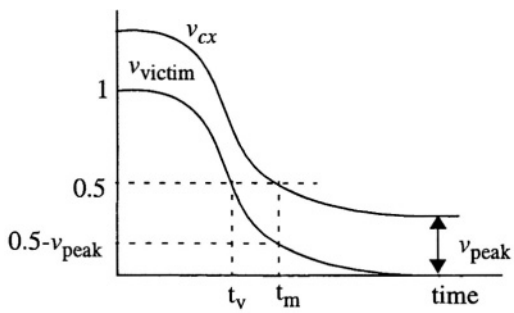


FIGURE 8.18 Waveforms for the proof for Theorem 8.2.

In some cases the peak of the noise bump may exceed the 50% switching threshold. In such a case, Theorem 8.2 is no longer valid and the circuit may cause a switching anytime after t_m. The circuit operation may be disrupted due to this glitch and it is indeed a reliability violation [8.12].

Theorem 8.3: *Define*

$$v_{cp}(t) = \sum_i v_{pi}(t) \tag{8.55}$$

$$v_{pi}(t) = \begin{cases} v_{\text{peak}i}, & t_{pi} < t < t_{qi} \\ 0, \text{elsewhere} \end{cases} \tag{8.56}$$

and $t_{qk} < t_{p(k+1)}$ *for all k, that is the pulses do not intersect. We refer to* $v_{cp}(t)$ *as the combo-pulse. Worst case 50% delay of* v_{cp} *is given by*

$$t_{50(vcp)} = \max(t_{50(vpi)}), \tag{8.57}$$

which corresponds to the maximum of the worst case 50% delays when pulses are considered separately. Moreover, a noise pulse can be divided into several sections with the same peak value and treated as a combo-pulse.

Proof. Since pulses do not intersect, we can write $v_c(t)$ as

$$v_c(t) = \begin{cases} v_{\text{victim}}(t) + v_{\text{peak}i}, & t_{pi} < t < t_{qi} \\ v_{\text{victim}}(t), \text{elsewhere} \end{cases} \tag{8.58}$$

The worst 50% crossing of v_c, $t_{50(vcp)}$, will occur in only one of the regions and is independent of the other pulses widths and heights. Therefore, we can solve for the worst case 50% delay for each v_{pi} individually and take the maximum among them. QED

Theorem 8.4: *Assume that the victim and aggressor switching windows are known. Let* t_{va}, t_{ma}, t_{vb} *and* t_{mb} *denote the special points on earliest and latest noise-free victim waveforms as shown in Figure 8.19. Similarly* t_{pa}, t_{qa}, t_{pb} *and* t_{qb} *denote the start and end points of the noise pulses corresponding to the aggressor's earliest and latest*

arrival times. The worst case 50% delay can be found from the interaction of the latest victim waveform with the enlarged noise pulse between t_{pa} and t_{qb}, that is

$$t_{50} = \begin{cases} min(t_{mb}, \max(t_{qb}, t_{vb})), & t_{pa} \le t_{mb} \\ t_{vb}, & t_{pa} > t_{mb} \end{cases}. \tag{8.59}$$

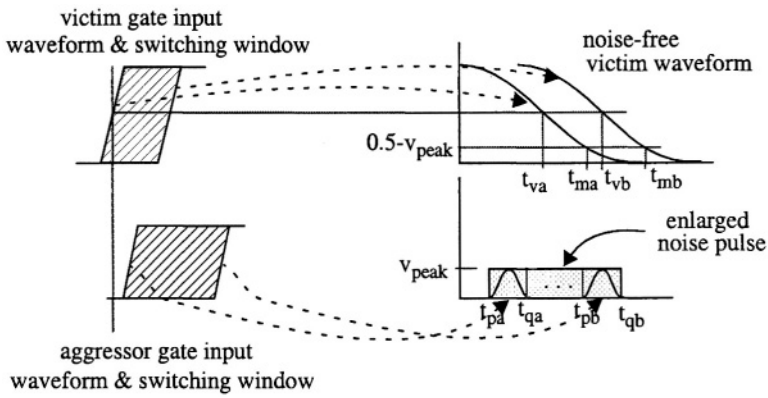


FIGURE 8.19 Enlarged noise pulse definition and application.

Proof. Enlargement of the noise pulse between t_{pa} and t_{qb} follows from the fact that the noise bump can be anywhere within the enlarged pulse, from the definition of a switching window. Since this enlarged noise pulse encapsulates any portion of it, from Theorem 8.3 its impact on 50% delay will be always worse than any portion of it. As for the worst case alignment of the victim waveform with the enlarged noise pulse, we must pick the latest one (i.e. t_{vb} and t_{mb}). Since t_{50} (worst alignment case 50% delay) is always between t_v and t_m according to Theorem 8.2, the larger they are, the larger the delay will be. t_{qb} is independent of the position of the victim waveform, therefore any victim waveform that happens before the latest one will yield less or equal t_{50}. QED

We can apply the theorems developed above to the multi-aggressor case. The noise pulses from each aggressor will be added to form a noise combo-pulse, as illustrated in Figure 8.20. The addition of noise pulses can be thought of as partitioning them into their intersecting portions and combining the pulses (Theorem 8.3). When combining the intersection sections, we add the heights of pulses to model the effect of

pulses at the same position from different aggressors. To solve for the worst case 50% delay, we apply Theorem 8.3 and Theorem 8.4. Namely, the worst case delays for each section in the combo-pulse are solved for using Theorem 8.4 and the results are combined using Theorem 8.3.

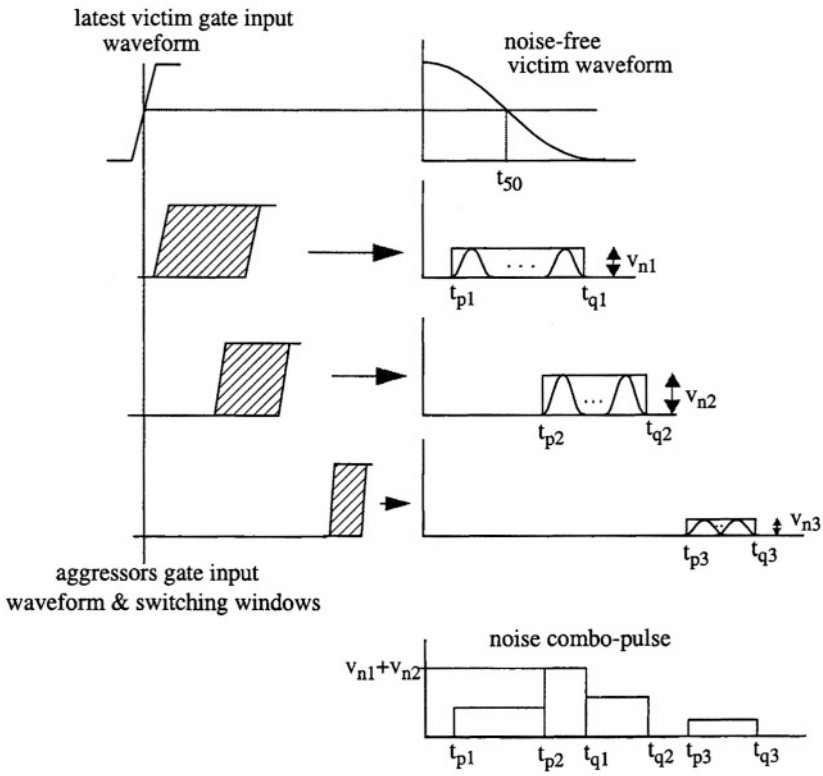


FIGURE 8.20 Extension of "noise pulse" concept for multiple aggressors.

It is easy to extend the noise pulse method to yield more accurate results by modeling the pulse to fit to the noise shape better. An illustration of an RLC coupling noise and its combo-pulse is illustrated in Figure 8.21.

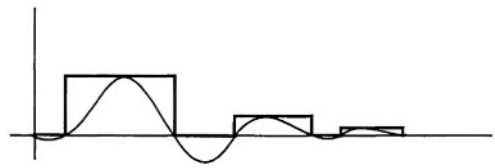


FIGURE 8.21 Non-unimodal noise pulses and its combo-pulse.

8.8.5 Computation details for coupled waveforms

This section is a generalization of the concepts in Section 8.6.6 to the multiport case. Different methods can be used to compute the currents for a multiport system. For efficiency reasons, moment based systems are more popular. It is better to abstract the coupled RC interconnect into driving point admittance and transfer function moments. The problem that we will consider here is as follows: How do we analyze the coupled linear circuit in Figure 8.22 which is obtained by combining the interconnect block with the Thevenin source models which are used to replace the gates?

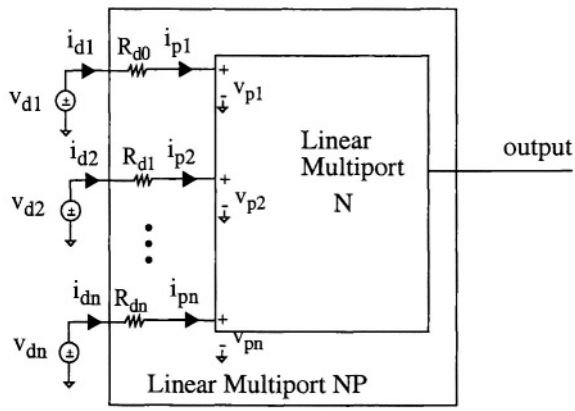


FIGURE 8.22 Combining Thevenin resistances with the multiport N.

Let $\mathbf{Y}(s)$ be the $n \times n$ admittance matrix for the interconnect portion of the circuit (multiport N in Figure 8.22). Thus

$$\mathbf{I}_p(s) = \mathbf{Y}(s)\mathbf{V}_p(s), \tag{8.60}$$

where $\mathbf{I}_p$ and $\mathbf{V}_p$ are, respectively, the vectors of the port voltages and currents of multiport N. Assume that moments of are $\mathbf{Y}(s)$ known:

$$\mathbf{Y}(s) = \mathbf{Y}_0 + \mathbf{Y}_1 s + \mathbf{Y}_2 s^2 + \ldots \tag{8.61}$$

Also let $\mathbf{H}(s)$ be the n-dimensional row vector containing the transfer functions from the multiport inputs to the output node, i.e.,

$$V_o(s) = \mathbf{H}(s)\mathbf{V}_p(s) \tag{8.62}$$

with the moment expansion

$$\mathbf{H}(s) = \mathbf{h}_0 + \mathbf{h}_1 s + \mathbf{h}_2 s^2 + \ldots \tag{8.63}$$

Now consider the multiport that is labeled NP in Figure 8.22. It is obtained by combining the coupled interconnect with the Thevenin driver models. From this model we would like obtain the admittance matrix, $\mathbf{Q}(s)$, and the transfer functions from Thevenin voltage sources to the output, $\mathbf{W}(s)$ to analyze the output and driving point responses.

Let $\mathbf{I}_d$ and $\mathbf{V}_d$ be the vectors of the port voltages and currents of multiport NP. Then

$$\mathbf{I}_d(s) = \mathbf{Q}(s)\mathbf{V}_d(s) \tag{8.64}$$

$$\mathbf{V}_d(s) - \mathbf{V}_p(s) = \mathbf{R}\mathbf{I}_d(s) = \mathbf{R}\mathbf{I}_p(s) \tag{8.65}$$

$$\mathbf{Q}(s) = (\mathbf{R} + \mathbf{Y}^{-1}(s))^{-1} \tag{8.66}$$

where $\mathbf{R}$ is the diagonal matrix such that R_{kk} is equal to the Thevenin resistance at port k. Applying the Bartlett-Sherman-Morrison-Woodbury formula [8.16] to (8.65) and dropping the s terms for convenience,

$$\mathbf{Q} = \mathbf{Y} - \mathbf{Y}\mathbf{R}(\mathbf{I} + \mathbf{Y}\mathbf{R})^{-1}\mathbf{Y} \tag{8.67}$$

Equation (8.67) nicely eliminates the inverse **Y** term. In coupled RC tree circuits, $\mathbf{Y}_0$ is equal to zero, therefore it is not invertible. Setting

$$\mathbf{B} = \mathbf{YR}(\mathbf{I} + \mathbf{YR})^{-1}, \tag{8.68}$$

the moments of **B** can be found using polynomial division as the following recursive formula:

$$\mathbf{B}_k = \mathbf{Y}_k - \sum_{i=0}^{k-1} \mathbf{B}\mathbf{Y}_{k-i}\mathbf{R}(\mathbf{I} + \mathbf{Y}_0\mathbf{R})^{-1}. \tag{8.69}$$

Inserting **B** into (8.66) and expanding moment terms, we can see that the moments of **Q** are given by

$$\mathbf{Q}_k = \mathbf{Y}_k - \sum_{i=0}^{k} \mathbf{B}_i\mathbf{Y}_{k-i} \tag{8.70}$$

After the moments of $\mathbf{Q}(s)$ are computed, it is straightforward to use AWE or other moment matching methods to find poles and residues. With the reduced order model contained dominant poles, the time domain responses can be obtained by applying symbolic convolution to solve for the driving point currents.

The transfer function moments at the output node of the victim net are updated similarly after the Thevenin resistances are connected to the multiport. It can be shown that

$$\mathbf{W}(s) = \mathbf{H}(s)(\mathbf{I} + \mathbf{RY})^{-1}. \tag{8.71}$$

Expanding the terms into polynomials and after some algebra, we obtain the following recursive formula to compute the moments of $\mathbf{W}(s)$:

$$\mathbf{w}_k = \left(\mathbf{h}_k - \sum_{i=0}^{k-1} \mathbf{w}_i\mathbf{R}\mathbf{Y}_{k-i}\right)(\mathbf{I} + \mathbf{RY}_0)^{-1}. \tag{8.72}$$

The *i*th element of the row vector $\mathbf{w}_k$ is the *k*th transfer function moment of the output node due to the *i*th Thevenin source.

8.9 Summary

Popular solutions as well as recent innovations are presented for the solution of the combined gate and interconnect system. The impact of crosstalk noise was also explored and methodologies were proposed to measure the delay and noise impact under a static timing analysis context. The reader should be warned that the gate delay modeling remains an ongoing research problem and may continue to evolve as new technologies emerge.

References

[8.1] R. B. Hitchcock, G. L. Smith, and D. D. Cheng, "Timing analysis of computer hardware," *IBM Journal of Research and Development,* vol. 26(1), pp. 100-105, January 1982.

[8.2] N. H. E. Weste and K. Eshraghian, *Principles of CMOS VLSI Design,* Second Edition. Reading, MA: Addison-Wesley, pp. 213, 1992.

[8.3] M. A. Horowitz, *Timing Models for MOS Circuits.* Ph. D. thesis, Stanford University, january 1984.

[8.4] N. Jouppi, "Timing analysis and performance improvement of MOS VLSI designs," *IEEE Trans. on Computer-Aided Design,* vol. CAD-6, pp. 650-665, 1987.

[8.5] J. K. Ousterhout, "A switch-level timing verifier for digital MOS VLSI," *IEEE Trans. on Computer-Aided Design,* vol. CAD-4, pp. 336-349, 1985.

[8.6] J. Qian, S. Pullela, and L. T. Pileggi, "Modeling the effective capacitance for the RC interconnect of CMOS gates," *IEEE Trans. on Computer-Aided Design,* vol. 13, no. 12, pp. 1526-1535, Dec. 1994

[8.7] R.E. Mains, T. A. Mosher, L.P.P.P. van Ginneken, and R.F. Damiano, "Timing verification and optimization for the PowerPC processor family," in *Proc. Intl. Conf. on Computer Design,* pp. 390-393, 1994.

[8.8] F. Dartu, N. Menezes, J. Qian, and L. T. Pillage, "A gate-delay model for high-speed CMOS circuits," in *Proc. 31st ACM/IEEE Design Automation Conference,* pp. 576-580, 1994.

[8.9] F. Dartu, N. Menezes, and L. T. Pileggi, "Performance computation for precharacterized CMOS gates with RC loads," *IEEE Trans. on Computer-Aided Design,* vol. 15, no. 5, pp. 544-553, May 1996.

[8.10] J. E. Bracken, "Passive modeling for linear interconnect networks," Technical Report, Dept. of ECE, Carnegie Mellon University, 1995.

[8.11] P. R. O'Brien and T. L. Savarino, "Modeling the driving point characteristic of resistive interconnect for accurate delay estimation," in *Proc. Int. Conf. Computer-Aided Design,* Nov. 1989.

[8.12] R. Arunachalam, F. Dartu, and L. T. Pileggi, "CMOS gate delay models for general RLC loading," in *Proc. International Conference on Computer Design,* 1997.

[8.13] G. Yee, R. Chandra, V. Ganesan, and C. Sechen, "Wire delay in the presence of crosstalk," In *Proceedings of the TAU Workshop on Timing in Digital Systems,* December, 1997.

[8.14] P. R. Gray and R. G. Meyer, *Analysis and Design of Analog Integrated Circuits,* Third Edition. Wiley Publishers, 1992.

[8.15] F. Dartu and L. T. Pileggi, "Calculating worst-case gate delay due to dominant capacitance coupling", in *Proc. IEEE/ACM Design Automation Conf.,* 1997.

[8.16] G. H. Golub and C. F. Van Loan, *Matrix Computations,* Third Edition. Baltimore: The Johns Hopkins University Press. 1996.

Index

A

Admittance
  block moments 190
  input 26
  matrix 186
Arnoldi
  approximation 231
  block 196–199, 216
  block, algorithm 197
  moment matching 198
Asymptotic waveform evaluation, see AWE
Attenuation 30, 105
  metric 106
AWE 17, 119–150, 189
  moment generation 122–125
  moment matching 126–128
  multipoint 149
  stability 132

B

Branch tearing 161–163

C

Capacitance
  coupling 3
  extraction 10
  FASTCAP 11
  multipole expansions 11
  pattern matching 7
Ceff 16, 275–292
Cholesky factorization 166, 227, 228, 245–246
Circuit compaction 163–166
Complex frequency hopping (CFH) 150
Congruence transformation 191, 206, 207
  split 204
Crosstalk 95

D

Damping 100
Delay metrics 25
  closed form 94
  Elmore delay 33
  gamma fitting 67–73
  h-gamma 73–80
  single-pole 35
  two-pole models 80–94
Dominant pole metric 35

E

Effective capacitance, see Ceff
Eigendecomposition 214, 246
Elmore delay 14, 25–57
  bound 46
  calculation 34
  definition 33
  for general input signals 49

F

Floating nodes 167–170
Frequency shifting 143–146, 176, 221

G

Gamma distribution 15, 68
Gate characterization 272–273
Gate delay 16, 271–304
  coupling 292–304
  effective capacitance 275
  k-factor equations 273
  pi-model 282
  RLC loading 289
  Thevenin modeling 277–292
  two-step approach 276
Gate resistance 284
Global wires 2
Graphs 154–156
  branches 154
  incidence matrix 155
  links 155
  nodes 154
  tree branches 155
  trees 155

H

H-gamma 73–80
  stability 78
  table lookup 79
Homogeneous response 73

I

Immittance functions 26
Impedance
  input 26
  matrix 186
Impulse response 28
Incidence matrix 155–156, 158
  reduced 155
Inductance 5, 9, 100
Inductor loops 167, 170
Interconnect
  electromagnetic modeling 9
  metrics 13
  models 6
  rc delay 31
  resistance 7
  scaling 1

K

Krylov subspaces 18, 191–202
Krylov vectors 192–193
  generation 224–226

L

Lanczos
  block 196, 200–202, 217
  block, algorithm 200
  breakdown 201
  look-ahead 201
  moment matching 202
  MPVL 202
  PVL 230
  SyMPVL 202
Local wires 2
LU factorization 123, 175, 177, 190, 191, 224
  forward and back substitution 175, 190

M

Macromodeling 18, 181
  circuit formulation 184
  using AWE 189
  using projection 196
Markov parameters 147
  matching 147–148
Maxwell's equations 9, 208

Mean 33, 46
Median 34, 46
Miller capacitance 293
MNA formulation 170–178, 184, 205, 265
  moment calculation 175
Modified Gram-Schmidt
    orthogonalization 219, 221
Moment matching 17, 126–132
  Arnoldi 198
  dominant pole convergence 138–139,
    143
  explicit 83, 137
  frequency scaling 137
  frequency shifting 143
  instability 132
  Lanzos 202
  moment matrix 128, 137
  moment shifting 140–143
  multipoint 149
  poles and residues 128
  PRIMA 212
Moments 39–43, 121–125
  calculation 41–43, 123–125, 153–178
  central 44–45, 58, 101
  circuit 12, 40
  probability 40
Multiport 182
  modeling 181–189
  port 182
  terminal 182

N

Newton-Raphson 94, 265
Noise metrics 95–100
  first moment 97
  two-pole 99
Nonlinear circuit simulation 19, 265

O

One-pole model 35, 80

P

Padé approximation 132–136, 231
  convergence 135
  diagonal sequence 135
  horizontal sequence 136
  instability 132, 134
  matrix 202, 229
  multipoint 150
  partial 149, 190
Padé table 135
Passivity 18, 202
  positive-realness 209
  preservation 209
Path tracing 156–170, 225
PEEC 9, 10, 208
Penfield and Rubinstein's bounds 37
Phase delay 30, 54, 105
  metric 110
Poles 27, 120, 128, 227
  dominant 121
  residues 27, 128, 227
PRIMA 18, 204–234
  block Arnoldi based algorithm 218
  circuit formulation 205
  error estimation 226, 236
  frequency shifting 221
  moment matching 212
  order selection 226
  passivity 209–211
  path tracing 225
  pole-residue representation 227
  post processing 227, 245–251
  projection 206
  RC circuits 228–230
  single-input multi-output 232
  state-space representation 227
Probability density function (p.d.f.) 33, 44,
    67, 100
  double exponential distribution 81
  gamma distribution 68
Projection 194, 194–196
  Arnoldi 198
  Lanczos 201
  oblique 194
  orthogonal 194
  Petrov-Galerkin conditions 194
  PRIMA 206
PVL 230, 236

PVL-WEB 236

Q

QR factorization 220

R

Ramp follower responses 52
Ramp response 29
RC circuits 10
RC trees 6, 31
  central moments 45
  delay 32
  driving point moments 57, 90, 113
  impulse responses 61
  moment calculation 41
  transfer functions 64
Recursive convolution 19, 227, 258–266
  numerical convolution 258
Residues 27
Resistance 7
Resistor loops 161–167
RICE 154, 166
RL circuits 10
RLC circuits 9, 10, 205, 208
  circuit formulation 188
RLC trees 8, 100, 106
  attenuation metric 106
  damping 104
  moment calculation 41, 153–161
  phase delay metric 110

S

S2P 90
Saturated ramp waveform 29
SPICE 19, 171, 208, 243
State-space
  equations 119
  state matrix 123
State-space realization
  from poles 251–255
  from PRIMA 245–249
  synthesis 256–257
Step response 28
Synthesis, equivalent circuit 19, 227, 256–257

T

Thevenin delay modeling 277–292
Transfer functions 25–31
  attenuation 30
  delay 30
  impulse response 28
  phase delay 30
  poles 27
  ramp response 29
  step response 28
  zeros 27
Transmission lines 100, 101, 177–179, 208
  moment generation 177
Two-pole models 15, 80–94
  delay calculation 94
  S2P 90
  stability 85
  table lookup 94
  with three moments 87

Y

Y parameters 183

Z

Z parameters 183
Zeros 27