Timing analysis of interconnect networks

Cao, Yi

2006

Cao, Y. (2006). Timing analysis of interconnect networks. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/35722>

<https://doi.org/10.32657/10356/35722>

Timing Analysis of Interconnect Networks

Cao Yi

School of Computer Engineering

A thesis submitted to the Nanyang Technological University
in fulfillment of the requirement for the degree of
Doctor of Philosophy

2006

Abstract

Timing analysis of high-order networks has been an important issue in system study. The delay is one of the important parameters to characterize a system and can be obtained from the transfer function of the system. With smaller feature sizes and increasing clock frequencies of today's semiconductor technology, interconnections between logic gates may actually be the dominant contributors of delay. In addition, interconnect effects such as ringing, reflection, crosstalk, dispersion and attenuation may corrupt logic signals and degrade system performance. These effects are not always handled appropriately, accurately or efficiently by the present levels of circuit simulators.

This thesis is to study and analyze the delay of interconnect structures and various networks containing large linear structures with nonlinear terminations. Approximation methods to compute delays of large interconnect networks are investigated. An approach for systematic interconnect design is to express the system behavior by poles and zeros. However, for a large interconnect network, computations of poles and zeros are extremely time-consuming. The time and accuracy performances of exact methods versus two approximate fast methods, the asymptotic waveform evaluation (AWE) technique and Padé approximation, are compared in evaluating the delays of circuits.

Since inductance makes peaking and oscillations possible, improperly designed interconnects can cause false switching and affect system reliability. Owing to the distributed nature of interconnect, poles of such systems are transcendental and large in number. Considering that the 2nd-order system is the simplest one that can reflect the effect of inductance, another approximation technique, namely the two-pole approximation method, is available for delay evaluation. This method uses the waveform of two oscillation parameters to find the two dominant approximate poles and thus obtains delay estimation. In this thesis, a new technique called the extended two-pole method is proposed to improve the accuracy of the above two-pole method by applying additional pairs of approximate poles.

All these approximation methods are efficient in approximating a large interconnect network by a reduced-order model. However, they inherently generate numerical instability

Abstract

which renders some computed poles unusable. It has been shown in this thesis that interpolated poles can be used to fill the gaps of missing poles and increase accuracy in evaluating delay.

The delay-time bound of a network has been a valuable theoretical and computational parameter in the design of analogue and digital circuits because of its fundamental importance in system performance. A basic model to derive the global switching time bound of a *RC*-ladder network in integrated circuits has been reported by other researchers. Since the *RLC*-cell is also commonly used in VLSI circuit design, similar delay-time bounds method is extended to *RLC*-ladder networks in this thesis. Furthermore, a method is also developed to compute the delay-time bounds of *RC*- and *RLC*-tree networks.

Despite the usefulness of the above delay-bound method, the obtained upper and lower bounds have been rather conservative; i.e. the range is rather wide. Thus, a novel algorithm has been developed in this thesis to refine the delay-time bounds of ladder and tree networks. It is based on using Lanczos process to obtain a reduced-order model of the state matrix of a network. The algorithm can generate an arbitrary-order model with little numerical degradation, but interestingly the network performance can usually be well described by just a 2nd-order model. With the Lanczos algorithm, the delay-time bound is closer to the actual delay time computed.

Finally, a few suggestions for future extension of this research work are that the techniques developed for delay evaluation in this thesis can be similarly applied to wave-digital interconnect networks and real computer networks. Since these networks have similar interconnect structures, their *RC*- or *RLC*-equivalent electric networks are development for timing analysis.

Acknowledgements

First of all I want to express my gratitude to my supervisor Dr. Tan Eng Chong for his invaluable guidance throughout this research work. He fostered and helped me shape the ideas presented in this thesis. Without his patience, understanding and encouragement, I cannot stick to and dip into my project, and the project would not be successful.

I would also like to thank the staff and technicians in the Center for Advanced Media Technology for their kind assistance in laboratory matter.

Finally, I want to thank all my friends for their support and encouragement.

Table of Contents

Abstract	i
Acknowledgements	iii
Table of Contents	iv
List of Figures	ix
List of Tables	xi
List of Acronyms and Abbreviations	xv
List of Symbols	xvi

Chapter 1: Introduction

1.1	Motivation	1
1.2	Exact Delay Computation	2
1.3	Approximate Delay Computation	2
1.4	Improved Approximate Delay Computation by Adding Extra Poles	3
1.5	Delay Time Bounds for RC - and RLC -Cell Networks	4
1.6	Optimized Delay Bound Estimation via Lanczos Process for RC - and RLC -Ladder Networks	4
1.7	Hardware and Software Platform	5

Chapter 2: Delay Evaluation

2.1	Introduction	6
2.2	Delay of a Network	6
2.2.1	Definition of Delay	6
2.2.2	Network Transfer Function	8
2.2.3	Step Response in Frequency and Time Domains	8
2.2.4	Delay Evaluation Using Step Response	10

Table of Contents

2.2.5	Algorithm for Delay Evaluation Using Step Response	11
2.3	Computation of Transfer Function	12
2.3.1	State-Space Method	12
2.3.2	Wave-Chain Method	15
2.4	Summary	17

Chapter 3: Computation of Approximate Delay

3.1	Introduction	18
3.2	Asymptotic Waveform Evaluation (AWE)	19
3.2.1	Moment Generation of AWE	19
3.2.2	Moment Matching of AWE	21
3.2.3	Pole-Residue Form of Approximate Transfer Function	23
3.2.4	Algorithm of AWE	24
3.3	Padé Approximation	25
3.3.1	Moment Generation of Padé Approximation	25
3.3.2	Moment Matching of Padé Approximation	25
3.3.3	Padé Table Analysis	26
3.3.4	Algorithm of Padé Approximation	29
3.4	Remedial Technique for AWE and Padé Approximation: Moment Scaling	30
3.5	Two-Pole Method	31
3.5.1	Equivalent Circuits of Transmission Lines	31
3.5.2	Moment Generation of Two-Pole Method	34
3.5.3	Generation of α Wave and β Wave	35
3.5.4	Two-Pole Approximation	35
3.5.5	Algorithm of Two-Pole Method	37
3.6	Approximate Delay Computation	38
3.7	Numerical Examples and Analysis	39
3.7.1	Delay Evaluation Using AWE Technique and Padé Approximation	39
3.7.2	Delay Evaluation Using Two-Pole Method	48
3.8	Summary	51

Chapter 4: Improvement of Approximate Delay Accuracy by the Addition of Extra Poles

4.1	Introduction	52
4.2	Improved AWE-Based Delay by Pole Interpolation	53
4.2.1	Generation of New Stable Poles Using Interpolation	53
4.2.2	Evaluation of New Approximate Delay	55
4.2.3	Algorithm of Improved AWE-Based Delay by Pole Interpolation	56
4.3	Delay Estimation by Extended Two-Pole Method	57
4.3.1	Generation of α Wave and β Wave	57
4.3.2	Extension of Two-Pole Approximation	58
4.3.3	Approximate Delay Computation	58
4.3.4	Algorithm of Delay Estimation by Extended Two-Pole Method	59
4.4	Numerical Examples and Analysis	60
4.4.1	Improved AWE-Based Delay by Pole Interpolation	60
4.4.2	Delay Evaluation Using Extended Two-Pole Method	72
4.5	Summary	77

Chapter 5: Delay Time Bounds for *RC*- and *RLC*-Cell Networks

5.1	Introduction	78
5.2	System Description	81
5.2.1	System Characterization of <i>RC</i> -Ladder Networks	81
5.2.2	System Characterization of <i>RC</i> -Tree Networks	82
5.2.3	System Characterization of <i>RLC</i> -Ladder Networks	83
5.2.4	System Characterization of <i>RLC</i> -Tree Networks	84
5.3	Derivation of Global Switching Time Bounds	86
5.3.1	Method to Compute Global Switching Time	86
5.3.2	Construction of Global Switching Time Bounds	87
5.3.3	<i>RC</i> -Ladder Networks	89
5.3.4	<i>RC</i> -Tree Networks	90

Table of Contents

5.3.5	<i>RLC</i> -Ladder Networks	92
5.3.6	<i>RLC</i> -Tree Networks	94
5.4	Delay Time Bounds of Interconnect Networks	98
5.5	Numerical Examples and Analysis of Delay Time Bounds	99
5.5.1	<i>RC</i> -Ladder Networks	99
5.5.2	<i>RC</i> -Tree Networks	99
5.5.3	<i>RLC</i> -Ladder Networks	102
5.5.4	<i>RLC</i> -Tree Networks	102
5.5.5	Analysis of Results	105
5.6	Summary	106

Chapter 6: Optimized Delay Bounds of *RC*-Ladder Networks via Lanczos Process

6.1	Introduction	107
6.2	Optimized Algorithm via Lanczos Process to Compute Global Switching Time Bounds of <i>RC</i> -Ladder Networks	107
6.2.1	System Reduction of Interconnect Networks	107
6.2.2	Delay Bound Connection of Lanczos Process	109
6.2.3	Reduced-Order Approximation via Lanczos Process	109
6.2.4	Approximate Eigenvalue and Eigenvector of <i>RC</i> -Ladder Networks via Lanczos Process	111
6.2.5	Improvement Proof of Eigenvalue and Eigenvector via Lanczos Process	116
6.2.6	Optimized Global Switching Time Bounds of <i>RC</i> -Ladder Networks via Lanczos Process	126
6.3	Delay Time Bounds of <i>RC</i> -Ladder Networks via Lanczos Process	128
6.4	Numerical Examples and Analysis	129
6.4.1	Another Method to Compute Improved Delay Time Bounds	129
6.4.2	Comparison of GST Bounds	130
6.4.3	Comparison of Delay Bounds	133
6.4.4	Approximation Order Selection in Lanczos Process	135
6.5	Summary	136

Chapter 7: Stopping Criteria of Delay Evaluation for *RLC*-Ladder Networks via Lanczos Process

7.1	Introduction	137
7.2	Optimized Algorithm via Lanczos Process to Compute Delay Time Bounds of <i>RLC</i> -Ladder Networks	137
7.2.1	Reduced-Order Approximation via Lanczos Process	137
7.2.2	Approximation of Lanczos Process	138
7.2.3	Optimized Global Switching Time Bounds of <i>RLC</i> -Ladder Networks via Lanczos Process	145
7.2.4	Delay Time Bounds of <i>RLC</i> -Ladder Networks via Lanczos Process	147
7.3	Numerical Examples and Analysis	147
7.4	Summary	152

Chapter 8: Conclusions and Future Work

8.1	Conclusions	153
8.2	Suggestions for Future Work	155
8.2.1	Timing Analysis of Wave-Digital Interconnect Networks	155
8.2.2	Timing Analysis of Computer Networks	156

List of Publications	165
-----------------------------	-----

References	166
-------------------	-----

List of Figures

Figure 2.1	Definition of delay of a circuit	7
Figure 2.2	Ladder networks	8
Figure 2.3	The equivalent replacement of elements: (a) Capacitor represented by a voltage source. (b) Inductor represented by a current source.	14
Figure 2.4	The model of a two-port system	15
Figure 3.1	Equivalent T circuit of a transmission line	33
Figure 3.2	Approximate equivalent circuit of a transmission line	33
Figure 3.3	Comparison of step responses using AWE, Padé and wave-chain methods (Butterworth low-pass filter, $N = 15$)	41
Figure 3.4	Comparison of step responses using AWE, Padé and wave-chain methods (Butterworth low-pass filter, $N = 30$)	42
Figure 3.5	Comparison of step responses using AWE, Padé and wave-chain methods (Chebyshev low-pass filter, $N = 15$ and $A_{\max} = 0.1\text{dB}$)	45
Figure 3.6	Comparison of step responses using AWE, Padé and wave-chain methods (Chebyshev low-pass filter, $N = 30$ and $A_{\max} = 0.1\text{dB}$)	46
Figure 3.7	Comparison of step responses using AWE, Padé and wave-chain methods (Bessel filter, $N = 8$ and $R_s = 1\Omega$)	48
Figure 3.8	Transmission line: $R_1 = 1.6\Omega$, $R_2 = 8.9\Omega$, $C_1 = 10\text{pF}$, $C_2 = 5\text{pF}$	48
Figure 3.9	The equivalent T-circuit	49
Figure 4.1	An n -cell (n^{th} -order) RC -tree network with capacitive load	63
Figure 4.2	Three interpolation curves joining useable approximate poles from the AWE method with $q=11$ (RC -tree network)	66
Figure 4.3	An n -cell ($2n^{\text{th}}$ -order) RLC -tree network with capacitive load	68
Figure 4.4	Three interpolation curves joining useable approximate poles obtained from the AWE method with $q=24$	70
Figure 4.5	Comparison of step responses using the state-space, AWE and two-pole methods (Two approximate poles are used)	73
Figure 4.6	Comparison of step responses using state-space, AWE, extended two-pole methods and PSpice (Six approximate poles are used)	76
Figure 5.1	An n -cell RC -ladder network with capacitive load	79

List of Figures

Figure 5.2	An n -cell RC -tree network with capacitive load (same as Figure 4.1 and repeated for convenience)	79
Figure 5.3	An n -cell RLC -ladder network with capacitive load	80
Figure 5.4	An n -cell RLC -tree network with capacitive load	80
Figure 6.1	An n -cell RC -ladder network with capacitive load (same as Figure 5.1 and repeated for convenience)	107
Figure 7.1	An n -cell RLC -ladder network with capacitive load (same as Figure 5.3 and repeated for convenience)	137
Figure 8.1	An Ethernet LAN with star topology	157
Figure 8.2	An Ethernet LAN with tree topology	157
Figure 8.3	A backbone network with mesh topology	158
Figure 8.4	A token ring LAN	158
Figure 8.5	Equivalent electrical network of the star network	159
Figure 8.6	Equivalent electrical network of the tree network	159
Figure 8.7	Equivalent electrical network of the mesh network	159
Figure 8.8	Equivalent electrical network of the token ring LAN	159
Figure 8.9	A WAN consisting of many network segments with different topologies	160
Figure 8.10	Equivalent electrical network of the WAN	161
Figure 8.11	A large LAN containing 4 long distance transmission links	162
Figure 8.12	Equivalent electrical network of LAN including long distance transmission links	163
Figure 8.13	Bifurcated routing network	164
Figure 8.14	Equivalent electrical network of the bifurcated routing network	164

List of Tables

Table 3.1	The Padé table	27
Table 3.2	Generalized element impedance and admittance	34
Table 3.3	Exact delays of Butterworth low-pass filters	40
Table 3.4	Approximate delays of Butterworth low-pass filters using the AWE technique	40
Table 3.5	Approximate delays of Butterworth low-pass filters using Padé approximation	41
Table 3.6	Exact delays of Chebyshev low-pass filters with $A_{\max} = 0.1$ dB	44
Table 3.7	Approximate delays of Chebyshev low-pass filters with $A_{\max} = 0.1$ dB using the AWE technique	44
Table 3.8	Approximate delays of Chebyshev low-pass filters with $A_{\max} = 0.1$ dB using Padé approximation	45
Table 3.9	Element values of Bessel filter	46
Table 3.10	Exact delays of Bessel filters with $R_s = 1\Omega$	47
Table 3.11	Approximate delays of Bessel filters with $R_s = 1\Omega$ using the AWE technique	47
Table 3.12	Approximate delays of Bessel filters with $R_s = 1\Omega$ using Padé approximation	47
Table 3.13	Moments and ratios (crest value is underlined)	50
Table 3.14	Approximate delays and execution time of two-pole method and AWE method	50
Table 4.1	Approximate poles and corresponding residues using the AWE method with $q = 10$ (Chebyshev low-pass filter, $N = 42$ and $A_{\max} = 0.1$ dB)	61
Table 4.2	Approximate poles and corresponding residues after Hermite-interpolation pole restoration (Chebyshev low-pass filter, $N = 42$ and $A_{\max} = 0.1$ dB)	62
Table 4.3	Approximate delays after pole restoration (Chebyshev low-pass filter, $N = 42$ and $A_{\max} = 0.1$ dB)	62
Table 4.4	Approximate poles of RC -tree network using the AWE method ($n=30$, $m = 2$, $\alpha = 0$, $\chi_1 = 14$ and $\chi_2 = 15$)	64

List of Tables

Table 4.5	Comparison of the exact delay and AWE approximate delays of the RC -tree network ($n=30$, $m=2$, $\alpha=0$, $\chi_1=14$ and $\chi_2=15$)	65
Table 4.6	Restored poles from different interpolation methods (RC -tree network; AWE method with $q=11$)	65
Table 4.7	Approximate delays by adding a part of restored poles into the usable AWE poles (RC -tree network; AWE method with $q=11$)	67
Table 4.8	Approximate delays by adding all restored poles into the usable AWE poles (RC -tree network; AWE method with $q=11$)	67
Table 4.9	Approximate poles of RLC -tree network using the AWE method ($n=20$, $m=2$, $\alpha=13$, $\chi_1=8$ and $\chi_2=11$)	69
Table 4.10	Comparison of the exact delay and AWE approximate delays of the RLC -tree network ($n=20$, $m=2$, $\alpha=13$, $\chi_1=8$ and $\chi_2=11$)	69
Table 4.11	Restored poles from different interpolation methods (RLC -tree network; AWE method with $q=24$)	70
Table 4.12	Approximate delays by adding a pair of restored poles into the usable AWE poles (RLC -tree network; AWE method with $q=24$)	71
Table 4.13	The exact delay and AWE approximate delays of the example interconnect network	72
Table 4.14	Approximate delay from different crests	72
Table 4.15	Approximate delays using more than one crest of the β wave	74
Table 4.16	AWE approximate poles and delay with different pole number	75
Table 4.17	Efficiency Comparison of AWE and extended two-pole methods	76
Table 5.1	Delay bounds for RC -ladder networks with $n=20$	100
Table 5.2	Delay bounds for RC -ladder networks with $n=30$	100
Table 5.3	Delay bounds for RC -tree networks with $n=5$, $m=2$ and $\chi_m=2$	100
Table 5.4	Delay bounds for RC -tree networks with $n=10$, $m=2$ and $\chi_m=7$	100
Table 5.5	Delay bounds for RC -tree networks with $n=15$, $m=2$ and $\chi_m=12$	101
Table 5.6	Delay bounds for RC -tree networks with $n=6$, $m=4$ and $\chi_m=2$	101
Table 5.7	Delay bounds for RC -tree networks with $n=10$, $m=4$ and $\chi_m=3$	101
Table 5.8	Delay bounds for RC -tree networks with $n=15$, $m=4$ and $\chi_m=5$	101
Table 5.9	Delay bounds for RLC -ladder networks with $u=1$ and $n=20$	102
Table 5.10	Delay bounds for RLC -ladder networks with $u=1$ and $n=25$	103

List of Tables

Table 5.11	Delay bounds for <i>RLC</i> -ladder networks with $u = 5$ and $n = 13$	103
Table 5.12	Delay bounds for <i>RLC</i> -ladder networks with $u = 5$ and $n = 15$	103
Table 5.13	Delay bounds for <i>RLC</i> -ladder networks with $u = 10$ and $n = 10$	103
Table 5.14	Delay bounds for <i>RLC</i> -ladder networks with $u = 10$ and $n = 12$	104
Table 5.15	Delay bounds for <i>RLC</i> -tree networks with $u = 1$, $m = 2$ and $\chi_m = 20$	104
Table 5.16	Delay bounds for <i>RLC</i> -tree networks with $u = 1$, $m = 3$ and $\chi_m = 20$	104
Table 5.17	Delay bounds for <i>RLC</i> -tree networks with $u = 5$, $m = 2$ and $\chi_m = 14$	104
Table 5.18	Delay bounds for <i>RLC</i> -tree networks with $u = 5$, $m = 3$ and $\chi_m = 14$	105
Table 5.19	Delay bounds for <i>RLC</i> -tree networks with $u = 10$, $m = 2$ and $\chi_m = 9$	105
Table 5.20	Delay bounds for <i>RLC</i> -tree networks with $u = 10$, $m = 3$ and $\chi_m = 9$	105
Table 6.1	Comparison of eigenvalue bounds for <i>RC</i> -ladder network with $\alpha = 2n$	131
Table 6.2	Comparison of GST bounds for <i>RC</i> -ladder network with $n = 3$ and $\lambda = 0.1$	131
Table 6.3	Comparison of GST bounds for <i>RC</i> -ladder network with $n = 3$ and $\lambda = 0.01$	131
Table 6.4	Comparison of GST bounds for <i>RC</i> -ladder network with $n = 5$ and $\lambda = 0.1$	132
Table 6.5	Comparison of GST bounds for <i>RC</i> -ladder network with $n = 5$ and $\lambda = 0.01$	132
Table 6.6	Comparison of GST bounds for <i>RC</i> -ladder network with $n = 9$ and $\lambda = 0.1$	132
Table 6.7	Comparison of GST bounds for <i>RC</i> -ladder network with $n = 9$ and $\lambda = 0.01$	133
Table 6.8	Comparison of delay bounds for <i>RC</i> -ladder network with $n = 20$	134
Table 6.9	Comparison of delay bounds for <i>RC</i> -ladder network with $n = 30$	134
Table 7.1	Comparison of eigenvalue and eigenvector bounds for <i>RLC</i> -ladder network with $u = 1$ and $n = 30$	148
Table 7.2	Comparison of eigenvalue and eigenvector bounds for <i>RLC</i> -ladder network with $u = 5$ and $n = 15$	148
Table 7.3	Comparison of eigenvalue and eigenvector bounds for <i>RLC</i> -ladder network with $u = 10$ and $n = 10$	148

List of Tables

Table 7.4	Comparison of delay bounds for <i>RLC</i> -ladder network with $u = 1$ and $n = 20$	149
Table 7.5	Comparison of delay bounds for <i>RLC</i> -ladder network with $u = 1$ and $n = 25$	149
Table 7.6	Comparison of delay bounds for <i>RLC</i> -ladder network with $u = 1$ and $n = 30$	149
Table 7.7	Comparison of delay bounds for <i>RLC</i> -ladder network with $u = 5$ and $n = 13$	150
Table 7.8	Comparison of delay bounds for <i>RLC</i> -ladder network with $u = 5$ and $n = 15$	150
Table 7.9	Comparison of delay bounds for <i>RLC</i> -ladder network with $u = 5$ and $n = 18$	150
Table 7.10	Comparison of delay bounds for <i>RLC</i> -ladder network with $u = 10$ and $n = 10$	151
Table 7.11	Comparison of delay bounds for <i>RLC</i> -ladder network with $u = 10$ and $n = 12$	151
Table 7.12	Comparison of delay bounds for <i>RLC</i> -ladder network with $u = 10$ and $n = 13$	151

List of Acronyms and Abbreviations

Acronym	Meaning
ATM	Asynchronous Transfer Mode
AWE	Asymptotic Waveform Evaluation
GST	Global Switching Time
IP	Internet Protocol
LAN	Local Area Network
LHP	Left-Half s -Plane
LP	Lanczos Process
OD	Origin/Destination
PA	Padé Approximation
RHP	Right-Half s -Plane
PVL	Padé Via Lanczos
RRP	Remedy by Restored Poles
VLSI	Very Large Scale Integration
WAN	Wide Area Network
WDF	Wave-digital filters

List of Symbols

Symbol	Meaning
α	Ratio of C_i and C
α_i	Real part of $\hat{p}_i$
α_m	Ratio of successive moments
$\alpha_{restored_i}$	Real part of $\hat{p}_{restored_i}$
β_i	Imaginary part of $\hat{p}_i$
β_m	Ratio of successive α_{m_j}
$\beta_{restored_i}$	Imaginary part of $\hat{p}_{restored_i}$
β_z	Imaginary part of $\hat{p}_z$
χ_k	Cell number of the k^{th} branch of tree network
ε	Ripple parameter in Chebyshev filter
ε_λ	Global switching time bound width
ε_d	Delay time bound width
ε_D	Delay time bound width
ε_k	The k^{th} moment skew
ε_i	Total moment skew
φ	Error rate of approximate delay
λ	Global switching time parameter
θ	Degree of the pole in polar coordinates
ω_n	Natural undamped frequency
ξ	Eigenvalue of $\hat{\mathbf{A}}_i^T$
ξ_S	Scaling factor
ζ	Damping ratio
Δ_m	Minute constant

List of Symbols

a_j	The j^{th} coefficient of $Q(s)$
$\hat{a}_j$	The j^{th} coefficient of $\hat{Q}(s)$
$A(s)$	Characteristic polynomial
A_i	Incident voltage wave
A_{max}	Maximum passband ripple attenuation of Chebyshev low-pass filter
$\mathbf{A}$	System state matrix
$\mathbf{A}_L$	Important matrix parameter in Lanczos algorithm
$\mathbf{A}_t$	$-\mathbf{A}$
$\hat{\mathbf{A}}_t^T$	Reduced-order approximation of $\mathbf{A}_t^T$
b_i	The i^{th} coefficient of $P(s)$
$\hat{b}_i$	The i^{th} coefficient of $\hat{P}(s)$
$\mathbf{b}$	One of state vectors of two-port system
B_i	Reflected voltage wave
$\mathbf{B}$	One of state matrices
$\hat{c}_j$	The j^{th} coefficient of $\hat{Q}_M(s)$
$\mathbf{c}$	One of state vectors of two-port system
C	Capacitor or Capacitance
C_{trans}	Capacitance per unit length of a transmission line
C_t	Capacitive load of interconnect network
C_T	Capacitance part of Z_T
$\mathbf{C}$	One of state matrices
d_{ij}	The i^{th} row and j^{th} column element of $\hat{\mathbf{A}}_t^T$
$\hat{d}_i$	The i^{th} coefficient of $\hat{P}_J(s)$
$D(s)$	Denominator polynomial of $Y_R(s)$
$\mathbf{D}$	One of state matrices
e	Number of data points to be interpolated
error	Modulus of error between the approximate and exact delays

List of Symbols

F	Matrix representing the contribution from memory elements
g	Number of restored poles obtained from useable pole interpolation
$g(x)$	Hermite interpolation polynomial (osculating polynomial)
G_{trans}	Conductivity per unit length of a transmission line
G	Matrix representing the contribution of memoryless elements
$h(t)$	Network transfer function in t -domain
$\hat{h}(t)$	Approximate network transfer function in t -domain
$H(s)$	Network transfer function in s -domain
$\hat{H}(s)$	Approximate network transfer function in s -domain
$\hat{H}(s)_{2p}$	Two-pole approximate transfer function
$\hat{H}_{new}(s)$	New approximate transfer function in s -domain
i_L	Inductor current
I	A unit matrix
J	Number of approximate zeors in $\mathcal{Q}_{[J/M]}$
J_{best}	Selected J that minimizes the total moment skew
k_{column}	The column sequence number of a matrix element
k_j	The j^{th} actual residue of $H(s)$
k_{row}	The row sequence number of a matrix element
k'_j	The j^{th} residue of $Y_R(s)$
$\hat{k}_i$	The i^{th} approximate residue
$l_j(x)$	j – order Lagrange interpolation polynomial
L	Inductor or Inductance
L_{trans}	Inductance per unit length of a transmission line
L_T	Inductance part of Z_T
m	Branch number of tree network
m_i	The i^{th} moment of $H(s)$

List of Symbols

$\hat{m}_i$	The i^{th} moment of $\hat{H}(s)$
M	Number of approximate poles in $Q_{[J/M]}$
M_{best}	Selected M that minimizes the total moment skew
n	Cell number of whole network
n_{sub}	Two-port subsystem number of an interconnect network
n'	Selected crest number in extended two pole method
N	Network order
$N(s)$	Numerator polynomial of $H(s)$
p_d	Order of derivative
p_j	The j^{th} actual pole of $H(s)$
$p_j(x)$	j – order interpolation polynomial
$[pole]_{\text{AWE}}$	The AWE approximate pole collection
$[pole]_{\text{RHP}}$	The pole collection in right-half s -plane
$[pole]_{\text{useable}}$	The useable approximate pole collection
$[pole]_{\text{restored}}$	The restored pole collection produced from pole interpolation
$[pole]_{\text{RRP}}$	The new pole collection after remedy by restored poles
$\hat{p}_i$	The i^{th} approximate pole
$\hat{p}_{\text{restored}_i}$	The i^{th} restored pole produced from pole interpolation
p'_j	The j^{th} pole of $Y_R(s)$
$\hat{p}_z$	A discarded pole that is in right-half s -plane
$P(s)$	Numerator polynomial of $H(s)$
$\hat{P}(s)$	Numerator polynomial of $\hat{H}(s)$
$\hat{P}_j(s)$	Numerator polynomial of $Q_{[J/M]}$
q	Approximate pole number
q_{ev}	Eigenvalue of $\mathbf{A}_i^T$
$\underline{q}_{\text{ev}}$	Lower bound of q_{ev}

List of Symbols

$\underline{q}_{ev_Hou}$	Lower bound of q_{ev} using the method given by Hou <i>et al.</i> (2000)
$\underline{q}_{ev_LP}$	Lower bound of q_{ev} generated by the method via Lanczos process
$\bar{q}_{ev}$	Upper bound of q_{ev}
$\bar{q}_{ev_Hou}$	Upper bound of q_{ev} using the method given by Hou <i>et al.</i> (2000)
$\bar{q}_{ev_LP}$	Upper bound of q_{ev} generated by the method via Lanczos process
$Q_{[J/M]}$	Padé approximant to $H(s)$ including J zeros and M poles
$Q(s)$	Denominator polynomial of $H(s)$
$\hat{Q}(s)$	Denominator polynomial of $\hat{H}(s)$
$\hat{Q}_M(s)$	Denominator polynomial of $Q_{[J/M]}$
r	Number of poles in the right-half s -plane
r_k	The k^{th} element of $\mathbf{r}$
r_p	Module of the pole in polar coordinates
$\underline{r}$	Lower bound of r_k
$\underline{r}_{LP}$	Lower bound of r_k generated by the method via Lanczos process
$\bar{r}$	Upper bound of r_k
$\bar{r}_{LP}$	Upper bound of r_k generated by the method via Lanczos process
$\mathbf{r}$	The eigenvector corresponding to q_{ev}
$\mathbf{r}_L$	Important vector parameter in Lanczos algorithm
$[residue]_{RRP}$	The corresponding residue collection to $[pole]_{RRP}$
R	Resistor or Resistance
R_l	Output resistance (resistor)
R_{trans}	Resistance per unit length of a transmission line
R_s	Input resistance (resistor)
R_T	Resistance part of Z_T
s	Frequency domain factor
s_0	An arbitrary, but fixed expansion point

List of Symbols

s_{ij}	The i^{th} -row j^{th} -column Component of $\mathbf{S}_i$
$s_j(x)$	The j – order Cubic-spline interpolation polynomial
$\mathbf{S}_i$	Scattering matrix
t	Time domain factor
t_{down}	Lower bound time
t_{final}	Enough large time
t_{ij}	The i^{th} -row j^{th} -column component of $\mathbf{T}_i$
t_{up}	Upper bound time
$(t_{22})_{\text{total}}$	The 2 nd -row 2 nd -column component of $\mathbf{T}_{\text{total}}$
T_λ	Global switching time
T_D	Network delay time
$\hat{T}_D$	Approximate network delay
$\hat{T}_{D_AWE}$	Approximate delay using the AWE technique
$\underline{T}_\lambda$	Lower bound of global switching time
$\underline{T}_{\lambda_Hou}$	Lower bound of T_λ using the method given by Hou <i>et al.</i> (2000)
$\underline{T}_{\lambda_LP}$	Lower bound of T_λ generated by the method via Lanczos process
$\underline{T}_D$	Lower bound of T_D
$\underline{T}_{D_Hou}$	Lower bound of T_D using the method given by Hou <i>et al.</i> (2000)
$\underline{T}_{D_LP}$	Lower bound of T_D generated by the method via Lanczos process
$\overline{T}_\lambda$	Upper bound of global switching time
$\overline{T}_{\lambda_Hou}$	Upper bound of T_λ using the method given by Hou <i>et al.</i> (2000)
$\overline{T}_D$	Upper bound of T_D
$\overline{T}_{D_Hou}$	Upper bound of T_D using the method given by Hou <i>et al.</i> (2000)
$\overline{T}_{D_LP}$	Upper bound of T_D generated by the method via Lanczos process
$\overline{T}_{\lambda_LP}$	Upper bound of T_λ generated by the method via Lanczos process

List of Symbols

$\mathbf{T}_i$	Wave-chain matrix
$\mathbf{T}_q$	Reduced-order approximation of $\mathbf{A}_{Lanczos}$ via Lanczos process
$\mathbf{T}_{total}$	Total wave-chain matrix
$u(t)$	Input variable in t -domain
$\mathbf{u}(t)$	Excitation vector in t -domain
$U(s)$	Input variable in s -domain
$\mathbf{U}(s)$	Excitation vector in s -domain
ν	Number of useable poles
ν_C	Capacitor voltage
ν_s	Voltage of input in t -domain
ν_t	Voltage of output in t -domain
ν_{gap}	Gap between $y_R(t)$ and $0.5y_{R_final}$
ν_{min}	Small enough constant
$\mathbf{v}_i$	The i^{th} moment vector of $\mathbf{X}(s)$
$\mathbf{v}_s$	Input voltage vector in t -domain
$\mathbf{v}_t$	Output voltage vector in t -domain
V_{column_lower}	Minimum of the sums of elements' values in all column of $\mathbf{A}_t^{-1}$
V_{column_upper}	Maximum of the sums of elements' values in all column of $\mathbf{A}_t^{-1}$
V_{row_lower}	Minimum of the sums of elements' values in all row of $\mathbf{A}_t^{-1}$
V_{row_upper}	Maximum of the sums of elements' values in all row of $\mathbf{A}_t^{-1}$
V_s	Voltage of input in s -domain
V_t	Voltage of output in s -domain
$\mathbf{x}_\xi$	The eigenvector corresponding to ξ
$\mathbf{x}(t)$	System state vector in t -domain
$\mathbf{x}^0$	Initial network state vector
$\mathbf{x}^\infty$	Steady network state vector

List of Symbols

$\mathbf{X}(s)$	System state vector in s -domain
$y(t)$	Output variable in t -domain
$y_R(t)$	Actual unit-step response in t -domain
y_{R_final}	Value of $y_R(t_{final})$
y_{trans}	The s -domain admittance per unit length of a transmission line
$\hat{y}_R(t)$	Approximate unit-step response in t -domain
$\hat{y}_{R_new}(t)$	New approximate unit-step approximate response in t -domain
$\mathbf{y}(t)$	Output vector in t -domain
$Y_{element}(s)$	Element Admittance
$Y_R(s)$	Actual unit-step response in s -domain
Y_{trans}	The s -domain admittance of a transmission line
Y_T	Admittance of the equivalent “T” circuit of a transmission line
$\hat{Y}_R(s)$	Approximate unit-step response in s -domain
$\hat{Y}_{R_new}(s)$	New approximate unit-step response in s -domain
$\mathbf{Y}(s)$	Output vector in s -domain
z_i	The i^{th} zero of $H(s)$
z_{trans}	The s -domain impedance per unit length of a transmission line
$\hat{z}_i$	The i^{th} zero of $\hat{H}(s)$
$\mathbf{z}(t)$	Network status vector
$\mathbf{z}^0$	Initial network status vector
$\mathbf{z}^\infty$	Steady network status vector
$Z_{element}(s)$	Element Impedance
Z_{trans}	The s -domain impedance of a transmission line
Z_T	Impedance of the equivalent “T” circuit of a transmission line

Chapter 1: Introduction

1.1 Motivation

With increasing demand for faster logic devices and higher capacity memories, the minimum-feature sizes in integrated circuit has widely been much scaled downward and VLSI circuit has included more and more complex interconnect structures. In today's deep submicron design, interconnect delay becomes the dominant factor, because it is the fundamental part of other techniques such as wire-sizing optimization (Chen *et al.* 1996, Chen and Wong 1996) and buffer insertion (Fishburn and Schevon 1995, Chen and Wong 1997).

Various techniques have been proposed for the delay analysis of interconnects. These techniques are based on either simulation techniques or analytical formulas. Simulation tools such as SPICE remain excellent tools for small size $RC(L)$ interconnect networks as they provide accurate and efficient estimation results. However, as the number of signal lines exceeds the million margins, in today's ICs, existing simulation tools become inefficiently time consuming to do the job. To address such a shortcoming, faster modeling methods based on moment-matching techniques are proposed, such as the AWE technique (Pillage and Rohrer 1990), Padé approximation (Chiprout and Nakhla 1994) and the two-pole approximation method (Kahng and Muddu 1994). Although the majority of those methods have a much faster estimation time than simulation tools such as SPICE, most of them may still remain numerically unstable under different conditions. Therefore, it is a timely and important research area to apply optimization method to current moment-matching methods for performance improvements.

In addition, delay bounds regarded as one of critical parameters in system timing analysis can be used to bound the delay, to bound the signal voltage and to certify if a circuit is "fast enough". Based on the importance of delay bounds in high-speed circuit design, it is desirable to find a better method to estimate a tighter delay bound if possible.

1.2 Exact Delay Computation

Because of its practical significance in analogue and digital circuits, the delay (Siebert 1994, Gupta *et al.* 1997, Celik *et al.* 2002) is regarded as one of the most important parameters in circuit analysis. The delay can adequately describe the performance of a network in the time-domain. For an interconnect network with step input, its delay, T_D , is defined as the instant when the transient step response would reach 50% of its final value (Kuo 1966, Siebert 1994, Celik *et al.* 2002).

To perform delay computation of an electronic circuit, its transfer function or its poles and zeros must be available. Two efficient methods, the state-space method (Carlson 1996) and the wave-chain (T-matrix) method (Ruston and Bordogna 1966, Chirlian 1969), can be used to derive the transfer function of the interconnect network in the frequency domain (s -domain). If the input is a step signal, the s -domain step response can be derived from the transfer function of the network. The corresponding step response in the time domain (t -domain) can then be obtained from the inverse Laplace transform, then its delay value can be obtained.

More details are given in Chapter 2.

1.3 Approximate Delay Computation

Circuit simulation tasks, such as the accurate prediction of interconnect effects at the board and chip level, or analog circuit analysis with full accounting of parasitic elements, may require the solution of large linear networks. These networks can become extremely large, especially when circuits are automatically extracted from layout, or contain models of distributed elements, such as transmission lines etc. For such large interconnect networks, it is generally impractical to compute their exact delay time because of the prohibitively large computation time. There exist some efficient algorithms that attempt to approximate the original system behavior with a lower order model.

The asymptotic waveform evaluation (AWE) technique (Pillage and Rohrer 1990,

Raghavan *et al.* 1993, Chiprout and Nakhla 1994) can be used to approximate the Laplace-domain transfer function of a linear network by a reduced-order model based on only a relatively small number of dominant poles and zeros. The reduced-order model is matched with the moments of the linear circuit, which are obtained from the Taylor series expansion of the circuit response around $s = 0$ (Maclaurin series). From the moments, the approximate dominant poles and zeros can be obtained and also the approximate transfer function can be computed.

The Padé approximation (Chiprout and Nakhla 1994, Gerald and Wheatley 1994, Baker and Graves-Morris 1996) which converts a power series into a rational function is another popular approach for finding dominant poles. By searching the Padé table to select the appropriate orders for the numerator and denominator polynomials, Padé approximation can get the best approximate transfer function. Thus, the transfer function of a high-order linear network can be approximated by a reduced-order model.

At high-enough frequencies, lumped elements are no longer adequate to describe the delay and distortion effects on interconnect lines. The two-pole approximation method (Kahng and Muddu 1994a,b, Vang and Dai 1994, Xue *et al.* 1994, Shao and Chen 2000) can be used to obtain two approximate dominant poles and corresponding residues of a interconnect network, which are expressed in terms of the moments of the network. A 2nd-order approximate transfer function can then be derived from the approximate dominant poles and residues.

More details are given in Chapter 3.

1.4 Improved Approximate Delay Computation by Adding Extra Poles

With increasing network order, the stability of the AWE method is in question. In high-order networks, the AWE method will produce some unstable approximate poles that are in the right-half s -plane (RHP) and should be discarded (Chiprout and Nakhla 1994). Therefore, the approximations may not achieve the desired accuracy. Based on the useable

approximate poles from the AWE, an interpolation method can be used to produce new approximate poles in the left-half s -plane (LHP). In this way, pole interpolation method can enhance the accuracy of delay computation.

The two-pole method is presented in Chapter 3 as an efficient but approximate method. To enhance the accuracy of delay evaluation using the two-pole method, a new technique called the extended two-pole method is proposed by using additional pairs of approximate dominant poles.

More details are given in Chapter 4.

1.5 Delay Time Bounds for *RC*- and *RLC*-Cell Networks

The delay time bound is an important performance indicator in interconnect circuits (Marinov and Neittaanm 1991, Marinov and Shivakumar 1996, Marinov 1998, Hou *et al.* 2000). Marinov and Shivakumar (1996) defined the global switching time (GST) and obtained the GST bounds of an *RC*-ladder network, a basic model used in integrated circuits. In this thesis, the estimation of GST bound is extended to *RC*-tree, *RLC*-ladder and *RLC*-tree networks. Furthermore, because the network delay time is a special case of the GST, the corresponding network delay bounds can be computed from the GST bounds obtained.

More details are given in Chapter 5.

1.6 Optimized Delay Time Bounds for *RC*- and *RLC*-Ladder Networks via Lanczos Process

A new estimation method for delay time bounds of *RC*- and *RLC*-ladder networks via the Lanczos process (Feldmann and Freund 1995) is proposed. It gives a tighter bound than the method presented in Section 1.5. In the new method, the Lanczos process is applied to generate a reduced-order model of the system matrix with little numerical degradation. In addition, using the Lanczos algorithm, the formulation of an *RC*- or *RLC*-ladder network can be described with a 2^{nd} -order model.

More details are given in Chapters 6 and 7.

1.7 Hardware and Software Platform

All the computations in this thesis are performed on a PC with Intel Celeron 1.1GHz CPU and Kingston 256M RAM, using MATLAB (Matlab 1996, Mathews and Fink 1999) v.6.5 application software. The hardware has 32 bits and the numerical results are generally accurate to 10^{-8} .

Chapter 2: Delay Evaluation

2.1 Introduction

The timing analysis of interconnect networks plays a vital role in tuning the performance of a system. The delay (Kuo 1966, Siebert 1994, Gupta *et al.* 1997, Celik *et al.* 2002) is one of the most important parameters in circuit performance, especially in high-order networks. The delay can be measured from the step response or impulse response. The delay from step response is applied here because it is the parameter which has been commonly used by engineers and researchers in this field (Pillage and Rohrer 1990).

The system properties of a circuit can be described by its transfer function. To compute the delay of an interconnect network, its transfer function must be derived first. The state-space method (Carlson 1996) and the wave-chain (T-matrix) method (Ruston and Bordogna 1966, Chirlian 1969) can be used to derive the transfer function of interconnect networks.

2.2 Delay of a Network

2.2.1 Definition of Delay

There are many definitions of delay time, T_D , given the actual transient response. The most straightforward definition is the time which the step response requires to reach 50% of its final value (Kuo 1966, Siebert 1994, Celik *et al.* 2002) as shown in Figure 2.1. The step response shown in Figure 2.1(a) is mainly for *RC*-ladder and *RC*-tree networks, and that shown in Figure 2.1(b) is mainly for *RLC*-ladder and *RLC*-tree networks. Especially, the step response in Figure 2.1(a) has little or no overshoot and this delay is also called as Elmore delay (Siebert 1994, Celik *et al.* 2002).

(a)

(b)

Figure 2.1 Definition of delay of a circuit

2.2.2 Network Transfer Function

Interconnect networks are typically modeled as linear networks. These networks generally consist of hundreds or thousands of components such as resistors, capacitors, inductors and transmission lines. The networks with only one input port and one output port can be regarded as two-port networks. An example is the structure shown in Figure 2.2 where R_s and R_l are the input resistor and output resistor, respectively.

The transfer function in s -domain is defined as

$$H(s) = \frac{V_t(s)}{V_s(s)} \quad (2.1)$$

The methods to compute the network transfer function are presented in Section 2.3.

Figure 2.2 Ladder networks

2.2.3 Step Response in Frequency and Time Domains

An interconnect network has the transfer function in the frequency domain, $H(s)$, which can be described as

$$\begin{aligned} H(s) &= \frac{P(s)}{Q(s)} = \frac{b_0 + b_1s + \dots + b_{N-1}s^{N-1}}{1 + a_1s + \dots + a_Ns^N} \\ &= \frac{k_1}{s - p_1} + \frac{k_2}{s - p_2} + \dots + \frac{k_j}{s - p_j} + \dots + \frac{k_N}{s - p_N} \end{aligned} \quad (2.2)$$

where a_i and b_i are coefficients, N is the network order (also the number of actual poles), p_j are the poles of $H(s)$, and k_j are the residues corresponding to p_j . More

details are shown in eqns. (3.10)-(3.12). Its unit-step response in the frequency domain, $Y_R(s)$, is given by

$$Y_R(s) = \frac{1}{s} \cdot H(s) \quad (2.3)$$

From eqn. (2.2), we have

$$Y_R(s) = \frac{N(s)}{D(s)} = \frac{H(s)}{s} = \frac{P(s)}{s \cdot Q(s)} = \frac{b_0 + b_1 s + \dots + b_{N-1} s^{N-1}}{s + a_1 s^2 + \dots + a_{N-1} s^N + a_N s^{N+1}} \quad (2.4)$$

And, $Y_R(s)$ can be expressed as partial fractions:

$$Y_R(s) = \frac{k'_1}{s - p'_1} + \frac{k'_2}{s - p'_2} + \dots + \frac{k'_j}{s - p'_j} + \dots + \frac{k'_N}{s - p'_N} + \frac{k'_{N+1}}{s - p'_{N+1}} \quad (2.5)$$

where the roots of $D(s)$, i.e. $p'_1, p'_2, \dots, p'_N, p'_{N+1}$, are the poles of $Y_R(s)$, and $k'_1, k'_2, \dots, k'_N, k'_{N+1}$ are the residues of $Y_R(s)$ at $s = p'_1, p'_2, \dots, p'_N, p'_{N+1}$, respectively.

In addition, comparing eqns. (2.2), (2.3) and (2.5), the relationship between p'_j and p_j is given by

$$p'_j = p_j \quad (1 \leq j \leq N) \quad (2.6)$$

and $p'_{N+1} = 0$. However, k'_j is totally different from k_j .

Applying inverse Laplace transform, the step response of the network in the time domain can be obtained as

$$y_R(t) = k'_1 e^{p'_1 t} + k'_2 e^{p'_2 t} + \dots + k'_N e^{p'_N t} + k'_{N+1} \quad t \geq 0 \quad (2.7)$$

where $e^{p'_{N+1} t} = 1$. Eqn. (2.7) is only for step response $Y_R(s)$ without repeated poles. If $Y_R(s)$ has repeated poles, there is a method (Carlson 1996) to get $y_R(t)$. However, the repeated poles can also be modified by adding a very small constant, Δ_m (eg. $\Delta_m = \text{eps}$ in Matlab (Matlab 1996) = 2^{-52}), to make $Y_R(s)$ include only distinct poles. If $Y_R(s)$ contains a pair of repeated poles, $D(s)$ can be expressed as

$$D(s) = (s - p'_1)(s - p'_2) \dots (s - p'_j)^2 (s - p'_{j+2}) \dots (s - p'_{N+1}) \quad (2.8)$$

Define $p'_{\text{replace } 1}$ and $p'_{\text{replace } 2}$ as: $p'_{\text{replace } 1} = p'_j - \Delta_m$ and $p'_{\text{replace } 2} = p'_j + \Delta_m$

respectively, where Δ_m is a small enough constant. Using $p'_{replace\ 1}$ and $p'_{replace\ 2}$ to replace p'_j and p'_{j+1} , $D(s)_{new}$ can be obtained. By denoting Δ as the error between $(s - p'_j)^2$ and $(s - p'_{replace\ 1})(s - p'_{replace\ 2})$, it can be derived that

$$\begin{aligned}\Delta &= (s - p'_j)^2 - (s - p'_{replace\ 1})(s - p'_{replace\ 2}) \\ &= (s - p'_j)^2 - [(s - p'_j)^2 - \Delta_m^2] = \Delta_m^2\end{aligned}\quad (2.9)$$

From eqn. (2.9), $D(s)_{new}$ will be close to $D(s)$ as long as Δ_m is small enough. In this way, eqn. (2.7) can be applied to get $y_R(t)$ again. The magnitude of the constant Δ_m is decided by the accuracy required.

2.2.4 Delay Evaluation Using Step Response

From the step response in the time domain, the delay can be computed according to the definition in Figure 2.1. The objective is to find the time-axis point at which the step response reaches 50 percent of its final value.

The value of $y_R(t)$ will gradually approach an asymptote which is its final value, y_{R_final} . This can be obtained by substituting a large enough value of t (say $t_{final} = 100,000s$), in $y_R(t)$. Since the transfer function of an interconnect network is usually complicated, the time for $y_R(t)$ to reach 50% of its final value cannot be obtained analytically by solving $y_R(t) = 0.5y_{R_final}$. Therefore, a numerical method is used and the value between $y_R(t)$ and $0.5y_{R_final}$ is defined as v_{gap} , i.e.

$$v_{gap} = y_R(t) - 0.5y_{R_final} \quad (2.10)$$

So that $y_R(t)$ can be presumed to reach $0.5y_{R_final}$ as long as v_{gap} is less than a given constant, v_{min} . When this is true, the corresponding t is the delay of the network and the magnitude of v_{min} can be used to decide the accuracy of the delay.

2.2.5 Algorithm for Delay Evaluation Using Step Response

The process in Sections 2.2.3 and 2.2.4 to obtain the delay of an interconnect network leads to the following algorithm.

Algorithm 2.1:

Step (0): Initialization: Give a large enough value for t_{up} , set $t_{down} = 0$ and set the values of t_{final} , Δ_m and v_{min} .

Step (1): Obtain the transfer function of an interconnect network, $H(s)$.

Step (2): The unit-step response in s -domain, $Y_R(s)$, can be obtained from eqn. (2.4) as

$$Y_R(s) = \frac{N(s)}{D(s)} = \frac{H(s)}{s} = \frac{P(s)}{s \cdot Q(s)} = \frac{b_0 + b_1s + \dots + b_{N-1}s^{N-1}}{s + a_1s^2 + \dots + a_{N-1}s^N + a_Ns^{N+1}}.$$

Step (3): Applying a root-solving algorithm on $D(s)$, it is can be obtained

$$\begin{aligned} D(s) &= s + a_1s^2 + \dots + a_{N-1}s^N + a_Ns^{N+1} \\ &= (s - p'_1)(s - p'_2) \dots (s - p'_j) \dots (s - p'_N)(s - p'_{N+1}) \end{aligned}$$

Thus, $Y_R(s)$ is easily to be expanded in partial-fraction form,

$$Y_R(s) = \frac{k'_1}{s - p'_1} + \frac{k'_2}{s - p'_2} + \dots + \frac{k'_j}{s - p'_j} + \dots + \frac{k'_N}{s - p'_N} + \frac{k'_{N+1}}{s - p'_{N+1}},$$

where p'_j and k'_j are the poles and residues of $Y_R(s)$, respectively.

(The AWE and Padé approximation method represented in Sections 3.2.2 and 3.3.2 are examples of solution methods for computing poles and residues.)

Step (4): If the poles have no repeated poles, go to Step (5);

else if, the poles include repeated poles, modify the repeated poles by adding small adjusting constants, $p'_{replace\ 1} = p'_j - \Delta_m$ and $p'_{replace\ 2} = p'_j + \Delta_m$.

Step (5): Compute the step response in t -domain,

$$y_R(t) = k'_1 e^{p'_1 t} + k'_2 e^{p'_2 t} + \dots + k'_{N+1} e^{p'_{N+1} t}.$$

Step (6): Compute the final value of $y_R(t)$, $y_{R_final} = y_R(t_{final})$.

Step (7): Compute the middle time, $t_{mid} = \frac{1}{2}(t_{up} + t_{down})$.

Step (8): Compute $v_{gap} = y_R(t_{mid}) - 0.5y_{R_final}$.

Step (9): If $v_{gap} < -v_{min}$, use t_{mid} to replace t_{up} . Then, compute new t_{mid} and new

$$v_{gap}, \quad t_{mid} = \frac{1}{2}(t_{up} + t_{down}), \quad v_{gap} = y_R(t_{mid}) - 0.5y_{R_final};$$

else if $v_{gap} \geq -v_{min}$, go to Step (10).

Step (10): If $v_{gap} > v_{min}$, use t_{mid} to replace t_{down} and go to Step (7);

else if $|v_{gap}| \leq v_{min}$, obtain the delay, $T_D = t_{mid}$. Exit.

In Step (3), the MATLAB (Matlab 1996, Fausett 1999) built-in function *residue()* is used to compute the poles and residues of the step response.

2.3 Computation of Transfer Function

In order to compute the delay of a network, the corresponding system transfer function is needed. For attaining the system transfer function, two methods are usually used: the state-space method (Carlson 1996) and the wave-chain (T-matrix) method (Ruston and Bordogna 1966, Chirlian 1969).

2.3.1 State-Space Method

The characteristics of an arbitrary circuit can be described in terms of two matrix equations, called the state equation and the output equation (Carlson 1996):

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t) \end{aligned} \tag{2.11}$$

where $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ and $\mathbf{D}$ are state-space matrices, $\mathbf{x}(t)$ is the N -dimensional state vector, $\mathbf{u}(t)$ the excitation vector, and $\mathbf{y}(t)$ the output vector. In the foregoing statement, N represents the number of energy-storage elements in the circuit such as capacitors and

inductors (and is equal to the order of the system).

Applying Laplace transforms to eqn. (2.11), we have

$$\begin{aligned} s\mathbf{X}(s) - \mathbf{x}(0^+) &= \mathbf{A}\mathbf{X}(s) + \mathbf{B}\mathbf{U}(s) \\ \mathbf{Y}(s) &= \mathbf{C}\mathbf{X}(s) + \mathbf{D}\mathbf{U}(s) \end{aligned} \quad (2.12)$$

where $\mathbf{X}(s)$ and $\mathbf{Y}(s)$ are respectively the state vector and the output vector in the s -domain. From eqn. (2.12), the transfer function in s -domain is obtained as

$$H(s) = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D} \quad (2.13)$$

where $\mathbf{I}$ is a unit matrix. The state equations and output equations can be expressed in the following two cases:

Case 1: $N = \text{odd}$

State equation:

$$\begin{bmatrix} v_{C_1}' \\ i_{L_2}' \\ v_{C_3}' \\ \vdots \\ v_{C_N}' \end{bmatrix} = \mathbf{A} \begin{bmatrix} v_{C_1} \\ i_{L_2} \\ v_{C_3} \\ \vdots \\ v_{C_N} \end{bmatrix} + \mathbf{B}\mathbf{v}_s$$

Output equation:

$$\mathbf{v}_t = \mathbf{C} \begin{bmatrix} v_{C_1} \\ i_{L_2} \\ v_{C_3} \\ \vdots \\ v_{C_N} \end{bmatrix} + \mathbf{D}\mathbf{v}_s \quad (2.14)$$

Case 2: $N = \text{even}$

State equation:

$$\begin{bmatrix} v_{C_1}' \\ i_{L_2}' \\ v_{C_3}' \\ \vdots \\ i_{L_N}' \end{bmatrix} = \mathbf{A} \begin{bmatrix} v_{C_1} \\ i_{L_2} \\ v_{C_3} \\ \vdots \\ i_{L_N} \end{bmatrix} + \mathbf{B}\mathbf{v}_s$$

Output equation:

$$\mathbf{v}_t = \mathbf{C} \begin{bmatrix} v_{C_1} \\ i_{L_2} \\ v_{C_3} \\ \vdots \\ i_{L_N} \end{bmatrix} + \mathbf{D}\mathbf{v}_s \quad (2.15)$$

where v_C and i_L , are respectively the capacitor voltage and the inductor current, $\mathbf{v}_t$ is the output voltage vector, and $\mathbf{v}_s$ is the input voltage vector.

The corresponding control sources shown in Figure 2.3 can be used to replace the elements in the circuit first. Then superposition is applied by suppressing all but one of independent sources to find the contribution from that source to each C_{v_C}' current, each

Li_L' voltage and each output of interest, and the process is repeated until all the independent sources are exhausted. Finally, based on eqns. (2.14) and (2.15), four state matrices can be obtained. Then, the transfer function of the interconnect network can be computed by using eqn. (2.13).

Figure 2.3 The equivalent replacement of elements: (a) Capacitor represented by a voltage source. (b) Inductor represented by a current source.

The above process to obtain the transfer function by the state-space method leads to the following algorithm:

Algorithm 2.2:

- Step (0): Initialization: Obtain the network order N , the value of the input resistor R_s , the value of the output resistor R_l and values of the various elements contained in every two-port subsystem.
- Step (1): Take the state variables to be all capacitor voltages and inductor currents, and mark their reference polarities on the circuit diagram.
- Step (2): Replace each capacitor with a fictitious voltage source and each inductor with a fictitious current source, labeled as in Figure 2.3.
- Step (3): Apply superposition by suppressing all but one of independent sources to find the contribution from that source to each Cv_c' current, each Li_L' voltage and each output of interest. Repeat for all other fictitious sources and independent input sources.

- Step (4): Sum the contributions to get complete expressions for each Cv_C' current and Li_L' voltage. Dividing these expressions by L or C yields the state equations. Obtain the output equations by summing the contributions to each output variable.
- Step (5): Compute **A**, **B**, **C** and **D** by considering the relation among state variable vectors, input vector, output vector and four state matrices. The relations are shown in eqns. (2.14) and (2.15).
- Step (6): Compute the transfer function of the network, $H(s) = C(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D}$. Exit.

2.3.2 Wave-Chain Method

Two-port ladder networks such as the networks shown in Figure 2.2 can be considered to consist of a series of two-port cascades (Ruston and Bordogna 1966) shown in Figure 2.4, in which A_1 and A_2 represent the incident voltage waves, B_1 and B_2 represent the reflected voltage waves, and R_1 and R_2 are the reference resistances at port 1 and port 2 respectively.

Figure 2.4 The model of a two-port system

They are related by

$$A_1 = V_1 + R_1 I_1, \quad A_2 = V_2 + R_2 I_2 \quad (2.16)$$

$$B_1 = V_1 - R_1 I_1, \quad B_2 = V_2 - R_2 I_2 \quad (2.17)$$

The transfer function is given by

$$H(s) = \frac{B_2}{A_1} \Big|_{A_2=0} \quad (2.18)$$

The scattering matrix, $\mathbf{S}_i = [s_{ij}]$, and the wave-chain matrix, $\mathbf{T}_i = [t_{ij}]$, of the i^{th} two-port component are defined as follows (Ruston and Bordogna 1966, Chirlian 1969):

$$\mathbf{S}_i = \begin{bmatrix} s_{11} & s_{12} \\ s_{21} & s_{22} \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} = \mathbf{S}_i \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} \quad (2.19)$$

$$\mathbf{T}_i = \begin{bmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} B_1 \\ A_1 \end{bmatrix} = \mathbf{T}_i \begin{bmatrix} A_2 \\ B_2 \end{bmatrix} \quad (2.20)$$

where

$$s_{11} = \frac{B_1}{A_1} \Big|_{A_2=0}, \quad s_{12} = \frac{B_1}{A_2} \Big|_{A_1=0}, \quad s_{21} = \frac{B_2}{A_1} \Big|_{A_2=0}, \quad s_{22} = \frac{B_2}{A_2} \Big|_{A_1=0} \quad (2.21)$$

$$t_{11} = s_{12} - s_{11}s_{22}/s_{21}, \quad t_{12} = s_{11}/s_{21}, \quad t_{21} = -s_{22}/s_{21}, \quad t_{22} = 1/s_{21} \quad (2.22)$$

From eqns. (2.18), (2.21) and (2.22), the transfer function of a two-port network is given by

$$H(s) = s_{21} = \frac{1}{t_{22}} \quad (2.23)$$

A two-port interconnect network can be seen as a system with n_{sub} sub-structures that are also two-port structures and have corresponding wave-chain and scattering matrices. According to the different subsystems of the interconnect network, the corresponding wave-chain matrices are denoted as $\mathbf{T}_i, (i \in [1, n_{\text{sub}}])$.

Hence, the total wave-chain matrix of the network is equal to the product of corresponding wave-chain matrices of individual two-port subsystems, i.e.

$$\mathbf{T}_{\text{total}} = \prod_{i=1}^{n_{\text{sub}}} \mathbf{T}_i \quad (2.24)$$

and the system transfer function is given by

$$H(s) = \frac{1}{(t_{22})_{\text{total}}} \quad (2.25)$$

in which $(t_{22})_{\text{total}}$ is the 2^{nd} row and 2^{nd} column element of $\mathbf{T}_{\text{total}}$.

The above process to obtain the transfer function of an interconnect network leads to the following algorithm.

Algorithm 2.3:

Step (0): Initialization: Obtain the network order number N , the value of input resistor R_s , the value of output resistor R_l and values of the various elements contained in every two-port subsystem.

Step (1): Compute the corresponding wave-chain matrix, $\mathbf{T}_m$, of every two-port subsystem.

Step (2): Compute the whole wave-chain matrix of the network, $\mathbf{T}_{total} = \mathbf{T}_1 \mathbf{T}_2 \cdots \mathbf{T}_{n-1} \mathbf{T}_n$.

Step (3): Compute $H(s) = (s_{21})_{total} = \frac{1}{(t_{22})_{total}}$. Exit.

2.4 Summary

As the minimum-feature sizes for integrated circuits scale downward, the delays have a significant, even dominant, impact on system performance. The delay is usually used for a linear system and defined as the time at which the monotonic step response reaches 50% of its final value. This definition has been widely used as an analytical estimate of interconnect delays in the performance-driven synthesis and layout of VLSI routing topologies.

The step-response of an interconnect network can be computed from its transfer function. Then the delay can be obtained. Two methods, the state-space and the wave-chain (T-matrix) methods, are introduced to compute the transfer function of the interconnect network.

For low-order interconnects, state-space method is good to generate very accurate calculation results. However, for high-order two-port systems, wave-chain method is more often used to combine subsystem transfer functions into the overall system transfer function due to its much better efficiency than state-space method.

Chapter 3: Computation of Approximate Delay

3.1 Introduction

Although the state-space and wave-chain methods can be used to compute the transfer function of an interconnect network, they are too time-consuming and lose stability as the order of the circuit increases. However, due to the distributed nature of interconnects, poles of such systems are transcendental and large in number. Two methods are investigated to approximate the actual system behaviour with a lower-order model. The two methods are the asymptotic waveform evaluation (AWE) technique (Huang *et al.* 1990, Pillage and Rohrer 1990, Raghavan *et al.* 1993, Chiprout and Nakhla 1994) and the Padé approximation (Chiprout and Nakhla 1994, Gerald and Wheatley 1994, Baker and Graves-Morris 1996).

The AWE technique seeks to capture the behaviour of a large linear circuit by approximating the Laplace-domain transfer function with a reduced-order model, containing only a relatively small number of dominant poles and zeros. The reduced-order model of the original system from the AWE technique is matched to the moments (Pillage and Rohrer 1994) of the original circuit and can be used to predict the time-domain or frequency-domain response of the linear network over a predetermined range of excitation frequencies. The necessity to solve the circuit at many discrete points is thus avoided and this results in a large saving of the computation and simulation time for a large interconnect network.

Padé approximation also seeks to capture the actual circuit behaviour by approximating the dominant poles of the circuit with a reduced-order model that is matched to the moments of the original circuit. At relatively higher frequencies, the AWE technique becomes inefficient and several methods are proposed to improve AWE's accuracy (Sungur *et al.* 1997). Padé approximation which converts a power series into a rational function is a popular approach utilized for finding dominant poles. It is used to approximate a

interconnect network with many poles by extracting dominant poles. The residue strength of the typical electric network poles is concentrated on a small region of the complex plane, making the approximation practical and accurate for these types of networks.

Numerical instability is one problem associated with the AWE technique and Padé approximation. A second problem is that both methods generate some unstable poles for passive systems. Although the first problem can be solved by using extended precision such as moment scaling method, the second problem is more difficult to handle. Anastasakis *et al.* (1994) proposed a method to enhance the stability by employing higher order moments.

Since the two-pole system is the simplest one that can reflect the effect of inductance, many researchers have attempted to capture the system behaviour using two-pole approximations (Sugiuchi *et al.* 1994, Kahng and Muddu 1994 a, b, Wang and Dai 1994, Zhou *et al.* 1994, Xue *et al.* 1996). An approach to find two approximate dominant poles and zeros were proposed by Shao and Chen (2000) and named two-pole method. A 2nd-order approximate transfer function can then be derived for delay estimation.

3.2 Asymptotic Waveform Evaluation (AWE)

The AWE technique consists of two main parts: moment generation and moment matching. It approximates the transient response of a circuit by first expanding the system equations in moments (coefficients) of a Taylor expansion about $s = 0$ (Maclaurin series) in the frequency domain, and then matching the initial boundary conditions and the first $2q$ moments (including the initial moment m_0) of the exact response to a low-order q -pole model (Huang *et al.* 1990, Pillage and Rohrer 1994).

3.2.1 Moment Generation of AWE

From the state-space form of interconnect network as given by eqn. (2.11), a single input and single output lumped, linear and time-invariant circuit can be described by the state equations,

Chapter 3: Computation of Approximate Delay

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{b}u(t) \\ y(t) &= \mathbf{c}^T \mathbf{x}(t) + du(t)\end{aligned}\tag{3.1}$$

where $\mathbf{x}(t)$ 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, $u(t)$ is the input variable, $y(t)$ is the output variable, and d is a scalar. Since the direct coupling term, d , can be computed separately, for simplicity, we will assume $d = 0$. Applying Laplace transforms to the state equations (3.1) above and with initial condition $\mathbf{x}(0) = \mathbf{0}$, it is derived that

$$\mathbf{X}(s) = (s\mathbf{I} - \mathbf{A})^{-1} \mathbf{b}U(s)\tag{3.2}$$

where $\mathbf{X}(s)$ and $U(s)$ are the Laplace transforms of $\mathbf{x}(t)$ and $u(t)$ respectively, and $\mathbf{I}$ is a unit matrix with $N \times N$ order. The transfer function of the circuit is thus given by

$$H(s) = \mathbf{c}^T (s\mathbf{I} - \mathbf{A})^{-1} \mathbf{b}\tag{3.3}$$

When the input is an impulse function, i.e. $U(s) = 1$, from (3.2) we have

$$\mathbf{X}(s) = (s\mathbf{I} - \mathbf{A})^{-1} \mathbf{b}\tag{3.4}$$

and it can be represented with a Taylor series expansion about $s = 0$ as

$$\mathbf{X}(s) = \mathbf{v}_0 + \mathbf{v}_1 s + \mathbf{v}_2 s^2 + \dots\tag{3.5}$$

where $\mathbf{v}_i$ is the i^{th} moment vector of $\mathbf{X}(s)$ and can be shown to be

$$\mathbf{v}_i = -\mathbf{A}^{-i-1} \mathbf{b}\tag{3.6}$$

which has a recursive relationship as

$$\begin{cases} \mathbf{v}_0 = -\mathbf{A}^{-1} \mathbf{b} \\ \mathbf{v}_i = -\mathbf{A}^{-1} \mathbf{v}_{i-1} \quad (i > 0) \end{cases}\tag{3.7}$$

$H(s)$ can also be represented with a Taylor series expansion as

$$H(s) = m_0 + sm_1 + s^2 m_2 + \dots + s^N m_N + \dots\tag{3.8}$$

where m_i is the i^{th} moment of the transfer function and can be calculated from

$$m_i = \mathbf{c}^T \mathbf{v}_i \quad (3.9)$$

These moments are the coefficients of Taylor expansion of the selected output node, given an impulse at the input.

3.2.2 Moment Matching of AWE

The transfer function of the circuit shown in eqn. (3.3) can also be described as

$$H(s) = \frac{P(s)}{Q(s)} = \frac{b_0 + b_1s + \dots + b_{N-1}s^{N-1}}{1 + a_1s + \dots + a_Ns^N} = \frac{\sum_{i=0}^{N-1} b_i s^i}{1 + \sum_{j=1}^N a_j s^j} \quad (3.10)$$

where $P(s)$ and $Q(s)$ are polynomials in s , and N is the number of actual poles (is also the network order). Applying partial fraction decomposition, the above equation is transformed into

$$H(s) = \frac{k_1}{s - p_1} + \frac{k_2}{s - p_2} + \dots + \frac{k_N}{s - p_N} = \sum_{j=1}^N \left(\frac{k_j}{s - p_j} \right) \quad (3.11)$$

where p_j and k_j are the poles and corresponding residues, respectively, of the circuit.

$H(s)$ may also be decomposed into $N-1$ zeros, z_i , as well as N poles as (Pillage and Rohrer 1994):

$$H(s) = K \frac{(s - z_1)(s - z_2) \dots (s - z_{N-1})}{(s - p_1)(s - p_2) \dots (s - p_N)} = K \frac{\prod_{i=1}^{N-1} (s - z_i)}{\prod_{j=1}^N (s - p_j)} \quad (3.12)$$

The form in eqn. (3.11) is more convenient for an inversion of Laplace transform into the time domain. The time domain impulse response of the output can be given in closed form in terms of the poles and residues as:

$$h(t) = \sum_{j=1}^N k_j e^{p_j t} \quad (3.13)$$

The poles, p_j , of $H(s)$ are the roots of the characteristic polynomial,

$$A(s) = \det(s\mathbf{I} - \mathbf{A}) \quad (3.14)$$

or the eigenvalues of the system matrix $\mathbf{A}$. For typical interconnect circuits, the number of poles can easily be on 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 is needed to seek is an efficient means to obtain those few dominant poles that adequately characterize circuit behaviour. One of the most effective procedures for approximating such a set of dominant poles (and zeros) is via moment matching.

Using the AWE technique, a low-order q -pole approximation to $H(s)$ can be found in one of the following forms (Pillage and Rohrer 1994):

$$\hat{H}(s) = \frac{\hat{P}(s)}{\hat{Q}(s)} = \frac{\hat{b}_0 + \hat{b}_1 s + \dots + \hat{b}_{q-1} s^{q-1}}{1 + \hat{a}_1 s + \dots + \hat{a}_q s^q} = \frac{\sum_{i=0}^{q-1} \hat{b}_i s^i}{1 + \sum_{j=1}^q \hat{a}_j s^j} = \frac{\prod_{i=1}^{q-1} (s - \hat{z}_i)}{\prod_{j=1}^q (s - \hat{p}_j)} \quad (q \ll N) \quad (3.15)$$

where $\hat{z}_j$ are the approximate zeros, $\hat{p}_j$ are the approximate poles, and the reduced pole number, q , is much less than the original number, N (i.e. $q \ll N$).

Consider the moment expansion of the reduced order model

$$\hat{H}(s) = \hat{m}_0 + \hat{m}_1 s + \hat{m}_2 s^2 + \dots \quad (3.16)$$

Then,

$$\frac{\hat{b}_0 + \hat{b}_1 s + \dots + \hat{b}_{q-1} s^{q-1}}{1 + \hat{a}_1 s + \dots + \hat{a}_q s^q} = \hat{m}_0 + \hat{m}_1 s + \dots + \hat{m}_{q-1} s^{q-1} + \hat{m}_q s^q + \dots + \hat{m}_{2q-1} s^{2q-1} \quad (3.17)$$

The AWE technique finds the coefficients of the q^{th} order reduced model by matching its first $2q$ moments to the first $2q$ moments of the original circuit function, i.e.

$$\hat{m}_i = m_i, \quad 0 \leq i < 2q \quad (3.18)$$

Thus,

$$\hat{b}_0 + \hat{b}_1 s + \dots + \hat{b}_{q-1} s^{q-1} = (m_0 + \dots + m_{2q-1} s^{2q-1})(1 + \hat{a}_1 s + \dots + \hat{a}_q s^q) \quad (3.19)$$

Equating coefficients of $s^q \rightarrow s^{2q-1}$ terms,

$$\begin{cases} 0 = m_q + m_{q-1}\hat{a}_1 + \cdots + m_0\hat{a}_q \\ 0 = m_{q+1} + m_q\hat{a}_1 + \cdots + m_1\hat{a}_q \\ \vdots \\ 0 = m_{2q-1} + m_{2q-2}\hat{a}_1 + \cdots + m_{q-1}\hat{a}_q \end{cases} \quad (3.20)$$

i.e.

$$\begin{bmatrix} m_{q-1} & m_{q-2} & \cdots & m_0 \\ m_q & m_{q-1} & \cdots & m_1 \\ \vdots & \vdots & \ddots & \vdots \\ m_{2q-2} & m_{2q-3} & \cdots & m_{q-1} \end{bmatrix} \begin{bmatrix} \hat{a}_1 \\ \hat{a}_2 \\ \vdots \\ \hat{a}_q \end{bmatrix} = - \begin{bmatrix} m_q \\ m_{q+1} \\ \vdots \\ m_{2q-1} \end{bmatrix} \quad (3.21)$$

from which $\hat{a}_i$ can be solved.

Equating coefficients of $s^0 \rightarrow s^{q-1}$ terms (Pillage and Rohrer 1994),

$$\begin{cases} \hat{b}_0 = m_0 \\ \hat{b}_1 = m_1 + m_0\hat{a}_1 \\ \vdots \\ \hat{b}_{q-1} = m_{q-1} + m_{q-2}\hat{a}_1 + \cdots + m_0\hat{a}_{q-1} \end{cases} \quad (3.22)$$

From the above equation, $\hat{b}_i$ can be solved. Thus, the approximate transfer function can be obtained from eqn. (3.15).

3.2.3 Pole-Residue Form of Approximate Transfer Function

Although the pole-zero representation is very useful in the analysis and design of analogue circuits, for time-domain interconnect analysis the following pole-residue representation is preferable:

$$\hat{H}(s) = \sum_{i=1}^q \frac{\hat{k}_i}{s - \hat{p}_i} \quad (3.23)$$

Upon solving $\hat{a}_i$, $\hat{p}_i$ can be computed from the denominator polynomial. The residues,

$\hat{k}_i$, can also be calculated directly from the moments. From the moment expansion,

$$\sum_{i=1}^q \frac{\hat{k}_i}{s - \hat{p}_i} = -\sum_{i=1}^q \frac{\hat{k}_i}{\hat{p}_i} - s \sum_{i=1}^q \frac{\hat{k}_i}{\hat{p}_i^2} - s^2 \sum_{i=1}^q \frac{\hat{k}_i}{\hat{p}_i^3} - \cdots \quad (3.24)$$

it is easily obtained that

$$\hat{m}_j = -\sum_{i=1}^q \frac{\hat{k}_i}{\hat{p}_i^{j+1}} \quad (3.25)$$

Matching the first q moments of the reduced model to those of the original circuit, we obtain a system of linear equations which can be solved for the residues (Pillage and Rohrer 1994):

$$\begin{bmatrix} \hat{p}_1^{-1} & \hat{p}_2^{-1} & \cdots & \hat{p}_q^{-1} \\ \hat{p}_1^{-2} & \hat{p}_2^{-2} & \cdots & \hat{p}_q^{-2} \\ \vdots & \vdots & & \vdots \\ \hat{p}_1^{-q} & \hat{p}_2^{-q} & \cdots & \hat{p}_q^{-q} \end{bmatrix} \begin{bmatrix} \hat{k}_1 \\ \hat{k}_2 \\ \vdots \\ \hat{k}_q \end{bmatrix} = - \begin{bmatrix} m_0 \\ m_1 \\ \vdots \\ m_{q-1} \end{bmatrix} \quad (3.26)$$

3.2.4 Algorithm of AWE

The above process to obtain the approximate transfer function of an interconnect network results in the following algorithm:

Algorithm 3.1:

Step (0): Initialization: Obtain the network order, N , the approximate pole number, q , the state matrix $\mathbf{A}$ and related state vectors $\mathbf{b}$ and $\mathbf{c}$.

Step (1): Compute the moments of the network,

$$m_i = \mathbf{c}^T \mathbf{v}_i, \quad (0 \leq i < 2q) \quad \text{where} \quad \mathbf{v}_i = -\mathbf{A}^{-i-1} \mathbf{b}.$$

Step (2): Compute the coefficients of $\hat{Q}(s)$, $\hat{a}_i$, by solving eqn. (3.21), $(1 \leq j \leq q)$.

Step (3): Compute the approximate poles, $\hat{p}_j$, by applying a root-solving algorithm on the resulting $\hat{Q}(s)$.

Step (4): Compute the approximate residues, $\hat{k}_j$, by solving eqn. (3.26), $(1 \leq j \leq q)$.

Step (5): Compute the approximate transfer function in s -domain,

$$\hat{H}(s) = \sum_{i=1}^q \frac{\hat{k}_i}{s - \hat{p}_i}. \text{ Exit.}$$

3.3 Padé Approximation

The AWE technique depends on the fixed number of approximate poles. However, the Padé approximation selects the best approximate number of poles and zeros from Padé table analysis (Chiprout and Nakhla 1994). The Padé approximation (PA) consists of three main parts: moment generation, moment matching and Padé table analysis.

3.5.2 Moment Generation of Padé Approximation

Padé approximation begins with a Maclaurin series expansion of the transfer function:

$$H(s) = m_0 + sm_1 + s^2m_2 + \cdots + s^Nm_N + \cdots \quad (3.27)$$

The Maclaurin moments contain information of the transfer function about the origin. In the frequency domain, these moments represent the properties of $H(s)$ in the low frequency range. The moment generation of Padé approximation is the same as that of the AWE technique as shown in Section 3.2.1. The moments of the system are also obtained from eqns. (3.6) and (3.9).

3.3.2 Moment Matching of Padé Approximation

Padé approximation employs a rational approximant. Padé approximant to $H(s)$ can be expressed as

$$\begin{aligned} Q_{[J/M]} = \hat{H}(s) &= \frac{\hat{P}_J(s)}{\hat{Q}_M(s)} = \frac{\hat{d}_0 + \hat{d}_1s + \cdots + \hat{d}_Js^J}{1 + \hat{c}_1s + \cdots + \hat{c}_Ms^M} = \frac{\sum_{i=0}^J \hat{d}_is^i}{1 + \sum_{j=1}^M \hat{c}_js^j} \\ &= \frac{(s - \hat{z}_1)(s - \hat{z}_2) \cdots (s - \hat{z}_J)}{(s - \hat{p}_1)(s - \hat{p}_2) \cdots (s - \hat{p}_M)} \end{aligned} \quad (3.28)$$

where M is the number of approximate poles, $\hat{p}_j$, and J is the number of approximate zeros, $\hat{z}_j$. Since there are $J+M$ unknown variables, $J+M$ moments are required to

be matched. The coefficients of $\hat{Q}_M$, $\hat{c}_j$, can be obtained from

$$\begin{bmatrix} m_{J-M+1} & m_{J-M+2} & \cdots & m_J \\ m_{J-M+2} & m_{J-M+3} & \cdots & m_{J+1} \\ \vdots & \vdots & \ddots & \vdots \\ m_J & m_{J+1} & \cdots & m_{J+M-1} \end{bmatrix} \begin{bmatrix} \hat{c}_M \\ \hat{c}_{M-1} \\ \vdots \\ \hat{c}_1 \end{bmatrix} = - \begin{bmatrix} m_{J+1} \\ m_{J+2} \\ \vdots \\ m_{J+M} \end{bmatrix} \quad (3.29)$$

The $\hat{p}_j$ can be obtained by applying a root-solving algorithm on the resulting $\hat{Q}_M$. The numerator coefficients, $\hat{d}_i$, can be found by equating powers $s^0, \dots, s^J$,

$$\begin{aligned} \hat{d}_0 &= m_0 \\ \hat{d}_1 &= m_1 + \hat{c}_1 m_0 \\ \hat{d}_2 &= m_2 + \hat{c}_1 m_1 + \hat{c}_2 m_0 \\ &\vdots \\ \hat{d}_J &= m_J + \sum_{j=1}^{\min(J,M)} \hat{c}_j m_{J-j} \end{aligned} \quad (3.30)$$

Thus, for various J and M , the corresponding approximate transfer functions, $Q_{[J/M]}$, can be obtained from eqn. (3.28).

An alternative approach is to factor the denominator in terms of its poles and then obtain the approximate residues of the approximate poles as eqn. (3.26). Applying partial fraction decomposition to eqn. (3.28), the following equation can be derived:

$$Q_{[J/M]} = \hat{H}(s) = \sum_{j=1}^M \frac{\hat{k}_j}{s - \hat{p}_j} \quad (3.31)$$

Thus, by choosing various J and M , the corresponding Padé approximants can also be obtained from the above equation.

3.3.3 Padé Table Analysis

For a high-order network, many possible choices of J and M exist. With different choices of $[J, M]$ approximation, it is possible to build an infinite array of the possible approximants. Our objective is to select an appropriate set of J and M which can make $Q_{[L/M]}$ as close to the exact transfer function as possible.

The Padé table shown in Table 3.1 is a two-dimensional infinite array of all approximants to a given Maclaurin series for different selections of J and M . Generally a diagonal or slightly off-diagonal Padé approximation is sufficient for realizing the maximum accuracy of Padé approximation. Hence, we can select J and M following the diagonal sequence when $J=M$ and an off-diagonal sequence when $J=M-1$.

Table 3.1 The Padé table

$J \rightarrow$ $M \downarrow$	0	1	2	3	...
0	[0/0]	[1/0]	[2/0]	[3/0]	...
1	[0/1]	[1/1]	[2/1]	[3/1]	...
2	[0/2]	[1/2]	[2/2]	[3/2]	...
3	[0/3]	[1/3]	[2/3]	[3/3]	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

The Padé approximation attempts to extract pole and residue information from the moments. It extracts far fewer approximate poles than actual poles in the network. The nature of Padé approximation is that it may produce both accurate and inaccurate poles in the approximation. The Padé approximation does not give good results when there exists poles farther away and of larger strength than those closer to the origin. Some poles that are far away from the expansion point are not accurate. Because many sets of approximate poles can be obtained from different values of J and M , some regulations are needed to choose appropriate J and M in the Padé table in order to get a series of approximate poles that make the approximate transfer function more accurate, and remove those inaccurate poles. In general, the selection of J and M is based on the following two criterions (Chiprout and Nakhla 1994).

(i) Criterion 1: A Padé table entry shown in eqn. (3.28) is eliminated as an approximation if the roots of its denominator does not include the $\{p_s\}$ (Chiprout and Nakhla 1994).

Justification:

The Padé approximation has an important property as follows:

$$\lim_{J \rightarrow \infty} Q_{[J/M]}(s) \rightarrow \prod_{j=1}^M \left(1 - \frac{s}{p_j} \right) \quad (3.32)$$

where p_j are the actual poles of the network arranged in order of increasing modulus:

$|p_1| \leq |p_2| \leq \dots \leq |p_M| \leq \dots$ and $|p_M| \neq |p_{M+1}|$. Using this property, the pole(s) closest to the expansion point, $s = 0$, can be obtained and labeled as $\{p_s\}$. Because the Padé approximation depends on the Maclaurin series expansion, its accuracy decreases with increased distance from the origin. If an approximation does not contain the $\{p_s\}$ as its roots, it is safe to assume that it is not an accurate approximation. Thus, Criterion 1 is derived. Thus, several table entries may be eliminated in this way without further processing. However, this criterion is not a sufficient condition for accuracy.

(ii) Criterion 2: Of the entries remaining from Criterion 1, the set of filtered (non-parasitic, left-half plane) poles and subsequent residues of each entry that minimizes the total moment skew, ε_t , is the one selected (Chiprout and Nakhla 1994).

Justification:

To make $\hat{H}(s)$ an approximation to $H(s)$, it can be derived that $\hat{m}_j$ is equal to m_j when j varies from 0 to $q-1$. That also means that there exists error between $\hat{m}_j$ and m_j when $j \geq q$. For a given moment m_k , and respective approximate moment $\hat{m}_k$, the k^{th} moment skew, ε_k , is defined as the scaled difference between the two values:

$$\varepsilon_k = \left| \frac{m_k - \hat{m}_k}{m_k} \right| \quad (3.33)$$

The total moment skew, ε_t , for a particular set of poles and residues is given by

$$\varepsilon_t = \sum_{k=q}^{\infty} \varepsilon_k \quad (3.34)$$

A large error in the approximation implies a large total moment skew. Thus, Criterion 2 is

obtained.

Through Criterion 1 and Criterion 2, the best J and M can be obtained, then the best approximant can be computed.

3.3.4 Algorithm of Padé Approximation

As represented above, the Padé approximation is used to compute the approximate transfer function of a two-port interconnect network. The process is given by the following algorithm.

Algorithm 3.2:

Step (0): Initialization: Obtain the network order, N , the approximate pole number, q , the state matrix $\mathbf{A}$ and related state vectors $\mathbf{b}$ and $\mathbf{c}$.

Step (1): Select a series of sets of J and M following the diagonal sequence when $J = M$ and an off-diagonal sequence when $J = M - 1$, ($M \leq N$).

Step (2): Compute the moments of the network,

$$m_i = \mathbf{c}^T \mathbf{v}_i, \quad (0 \leq i < 2q) \quad \text{where} \quad \mathbf{v}_i = -\mathbf{A}^{-i-1} \mathbf{b}, \quad (0 \leq i \leq J + M).$$

Step (3): Compute the coefficients of $\hat{Q}_M$, $\hat{c}_j$, by solving eqn. (3.29), ($1 \leq j \leq M$).

Step (4): Compute the coefficients of $\hat{P}_J$, $\hat{d}_i$, from eqn. (3.30), ($1 \leq i \leq J$).

Step (5): Compute the approximate transfer functions with all sets of J and M obtained in Step (1),

$$Q_{[J/M]} = \hat{H}(s) = \frac{\hat{P}_J(s)}{\hat{Q}_M(s)} = \frac{\hat{d}_0 + \hat{d}_1 s + \cdots + \hat{d}_J s^J}{1 + \hat{c}_1 s + \cdots + \hat{c}_M s^M} = \frac{(s - \hat{z}_1)(s - \hat{z}_2) \cdots (s - \hat{z}_J)}{(s - \hat{p}_1)(s - \hat{p}_2) \cdots (s - \hat{p}_M)}.$$

Step (6): Compute $\{p_s\}$ from eqn. (3.32).

Step (7): Substitute the $\{p_s\}$ into $\hat{Q}_M(s)$ obtained in Step (5). Then, only keep the corresponding sets of J and M to make the residual error $\hat{Q}_M(p_s)$ is zero or sufficiently small.

Step (8): With various sets of J and M left in Step (7), compute the total moment

skew, ε , from eqns. (3.33) and (3.34).

Step (9): Select the set of J and M that minimizes the total moment skew, and denote them as J_{best} and M_{best} .

Step (10): Compute the approximate transfer function, $Q_{[J_{best}/M_{best}]}$. Exit.

3.4 Remedial Technique for AWE and Padé Approximation: Moment Scaling

For a RLC -cell network, when the given values of inductances and capacitances are very small compared to the order of the values of resistances, the moments, as a result, will display a large shift of order between one moment and the next. It may produce ill-conditioned moment matrix and may lead to inaccurate results. To rectify this problem, moment scaling is useful. The idea is to replace the original moments m_k by $\xi_s^k m_k$, $k = 0, 1, \dots, 2q-1$, where $\xi_s > 0$ is a suitably chosen scaling factor. Thus, a frequency scaling of ξ_s is:

$$H(\xi_s s) = m_0 + (\xi_s m_1)s + (\xi_s^2 m_2)s^2 + \dots + (\xi_s^N m_n)s^N + \dots$$

There are three natural choices for scaling factors:

$$\xi_{s1} = \frac{1}{\|\mathbf{A}\|_2}, \quad \xi_{s2} = \frac{|m_0|}{|m_{2q-1}|}, \quad \text{and} \quad \xi_{s3} = \left(\frac{|m_0|}{|m_{2q-1}|} \right)^{1/(2q-1)} \quad (3.35)$$

The first choice (Anastakis *et al.* 1993) in eqn. (3.35) corresponds to scaling the matrix $\mathbf{A}$, which is one of four state-space matrices, to have Euclidean norm 1. The second choice (Huang 1990) is such that the two first scaled moments, m_0 and $\xi_{s2} m_1$, have the same magnitude. The third choice (Chiprout and Nakhla 1994), which is mainly used in this thesis, is such that, after scaling, the first and the last computed moment, m_0 and $\xi_{s3}^{2q-1} m_{2q-1}$, have the same magnitude. Hence, the result is a frequency scaling of ξ_{s3} :

$$H(\xi_{s3} s) = m_0 + (\xi_{s3} m_1)s + (\xi_{s3}^2 m_2)s^2 + \dots + (\xi_{s3}^N m_n)s^N + \dots \quad (3.36)$$

In other words, the new moments after scaling are

$$m_i = \xi_{s3}^i m_i' \quad (i \geq 0) \quad (3.37)$$

From eqn. (3.33), it is found that the transfer function obtained from the scaled moments is not accurate, and the transform from $H(\xi_{s3}s)$ to $H(s)$ is necessary.

3.5 Two-Pole Method

Since the two-pole system is the simplest one that can reflect the original system performance, a new approach called two-pole method (Xue *et al.* 1994, Shao and Chen 2000) that is different from the present approximation methods to create a two-pole approximate model is proposed. It finds two approximate dominant poles and zeros which are calculated by using moments. The output response is characterized by two parameters, and the objective function and constraints, as functions of these two parameters, form a constrained multivariable nonlinear optimization problem. In Section 3.2.1, an approach has been introduced to generate circuit moments. Thus, after obtaining two approximate poles and residues, the approximate transfer function of the circuit can be computed.

3.5.1 Equivalent Circuits of Transmission Lines

When compare to logic gates, interconnects play a major role on signal delay in high performance systems where the inductance effect becomes significant. Owing to the small cross-sections of on-chip interconnections and thin film package wires, they have significant resistance. At high-enough frequencies, lumped elements are no longer accurate enough to adequately describe the delay and distortion effects on interconnect lines. It has been shown that interconnects of thin film package behave like lossy transmission lines (Frye and Chen 1992).

Since interconnects are modeled as lossy transmission lines to reflect the effect of inductance in high performance systems, some properties of transmission lines should be considered in computing the exact and approximate transfer function of interconnects. By

Chapter 3: Computation of Approximate Delay

constructing the circuit using lumped circuit elements, there is a model to make studies on the transmission line for varying conditions. For a two-port transmission line, its impedance and admittance per unit length in s -domain can be expressed as

$$\begin{aligned} z_{trans} &= R_{trans} + j\omega L_{trans} = R_{trans} + sL_{trans} \\ y_{trans} &= G_{trans} + j\omega C_{trans} = G_{trans} + sC_{trans} \end{aligned} \quad (3.38)$$

where the line parameters: R_{trans} , G_{trans} , L_{trans} , C_{trans} , are the resistance, conductance, inductance and capacitance per unit length of the transmission line respectively, and l_{trans} is the length of the corresponding line. Thus, we have

$$\begin{aligned} Z_{trans} &= z_{trans} l_{trans} = l_{trans} R_{trans} + s l_{trans} L_{trans} \\ Y_{trans} &= y_{trans} l_{trans} = l_{trans} G_{trans} + s l_{trans} C_{trans} \end{aligned} \quad (3.39)$$

where Z_{trans} and Y_{trans} are the impedance and admittance of the transmission line in s -domain, respectively.

An equivalent “T” circuit (Anderson 1985, Gardiol 1987) can be found for describing a transmission line as shown in Figure 3.1 in which

$$\frac{Z_T}{2} = \frac{Z_{trans}}{2} \frac{\tanh \frac{\sqrt{Z_{trans} Y_{trans}}}{2}}{\frac{\sqrt{Z_{trans} Y_{trans}}}{2}} \quad \text{and} \quad Y_T = Y_{trans} \frac{\sinh \frac{\sqrt{Z_{trans} Y_{trans}}}{2}}{\frac{\sqrt{Z_{trans} Y_{trans}}}{2}} \quad (3.40)$$

Figure 3.1 Equivalent T circuit of a transmission line

Since

$$\sinh x = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \frac{x^7}{7!} + \dots \quad (3.41)$$

$$\tanh x = x - \frac{1}{3}x^3 + \frac{2}{15}x^5 + \dots \quad (3.42)$$

Chapter 3: Computation of Approximate Delay

and considering that if only the first two terms of the series are used for reasonable accuracy, it is derived that

$$\frac{\sinh \sqrt{Z_{trans} Y_{trans}}}{\sqrt{Z_{trans} Y_{trans}}} = 1 + \frac{Z_{trans} Y_{trans}}{3!} + \frac{(Z_{trans} Y_{trans})^2}{5!} + \dots \approx 1 + \frac{Z_{trans} Y_{trans}}{3!} \quad (3.43)$$

$$\frac{\tanh \frac{\sqrt{Z_{trans} Y_{trans}}}{2}}{\frac{\sqrt{Z_{trans} Y_{trans}}}{2}} = 1 - \frac{Z_{trans} Y_{trans}}{12} + \frac{(Z_{trans} Y_{trans})^2}{120} - \dots \approx 1 - \frac{Z_{trans} Y_{trans}}{12} \quad (3.44)$$

Considering a transmission line with $G_{trans} = 0$ and small parameter values, eqn. (3.40) can be simplified as

$$\begin{aligned} Y_T &= Y_{trans} \cdot \frac{\sinh \sqrt{Z_{trans} Y_{trans}}}{\sqrt{Z_{trans} Y_{trans}}} \approx s l_{trans} C_{trans} \left(1 + \frac{Z_{trans} Y_{trans}}{3!}\right) \\ &= s l_{trans} C_{trans} + \frac{s^2 l_{trans}^3 C_{trans}^2 R_{trans} + s^3 l_{trans}^3 C_{trans}^2 L_{trans}}{6} \approx s l_{trans} C_{trans} \end{aligned} \quad (3.45)$$

$$\begin{aligned} \frac{Z_T}{2} &= \frac{Z_{trans}}{2} \cdot \frac{\tanh \frac{\sqrt{Z_{trans} Y_{trans}}}{2}}{\frac{\sqrt{Z_{trans} Y_{trans}}}{2}} \approx \frac{Z_{trans}}{2} \left(1 - \frac{Z_{trans} Y_{trans}}{12}\right) \\ &= \frac{l_{trans} R_{trans}}{2} + \left(\frac{l_{trans} L_{trans}}{2} - \frac{l_{trans}^3 R_{trans}^2 C_{trans}}{24}\right) s - \frac{l_{trans}^3 R_{trans} L_{trans} C_{trans}}{12} s^2 - \frac{l_{trans}^3 L_{trans}^2 C_{trans}}{24} s^3 \\ &\approx \frac{l_{trans} R_{trans}}{2} + \left(\frac{l_{trans} L_{trans}}{2} - \frac{l_{trans}^3 R_{trans}^2 C_{trans}}{24}\right) s \end{aligned} \quad (3.46)$$

Thus, the approximate equivalent circuit of a transmission line can be described by using only lumped elements, as shown in Figure 3.2.

Figure 3.2 Approximate equivalent circuit of a transmission line

Table 3.2 Generalized element impedance and admittance

Element	$Z_{element}(s)$	$Y_{element}(s)$
Resistor	R	$\frac{1}{R}$
Inductor	sL	$\frac{1}{sL}$
Capacitor	$\frac{1}{sC}$	sC

Since the expressions for $Z_{element}(s)$ and $Y_{element}(s)$ for resistors, inductors and capacitors have the same form as those listed in Table 3.2., the values of R_T , L_T and C_T in Figure 3.2 can be easily computed from eqns. (3.45) and (3.46) as

$$R_T \approx \frac{l_{trans} R_{trans}}{2}, \quad L_T \approx \frac{l_{trans} L_{trans}}{2} - \frac{l_{trans}^3 R_{trans}^2 C_{trans}}{24} \quad \text{and} \quad C_T \approx l_{trans} C_{trans} \quad (3.47)$$

3.5.2 Moment Generation of Two-Pole Method

By using the approximate equivalent circuit shown in Figure 3.2 to replace transmission lines, an interconnect network at high frequency, that are modeled to contain lossy transmission lines, can be described by an equivalent network to include only the lumped elements.

As stated in Section 3.2.1, the transfer function of a lumped system is given by

$$H(s) = \mathbf{c}^T (s\mathbf{I} - \mathbf{A})^{-1} \mathbf{b} \quad (3.3)$$

Then, $H(s)$ can also be represented with a Taylor series expansion as

$$H(s) = m_0 + sm_1 + s^2 m_2 + \cdots + s^N m_N + \cdots \quad (3.8)$$

where m_i is the i^{th} moment of the transfer function. The detailed method to generate these moments has been described in Section 3.2.1. The moments can be obtained from eqns. (3.6) and (3.9).

3.5.3 Generation of α Wave and β Wave

After obtaining the moments of the network, two parameters that are functions of the moments can be computed. Let α_m represent the ratio of successive moments and β_m represent the ratio of successive α_m 's:

$$\alpha_{m-j} \equiv \frac{m_j}{m_{j+1}} \quad \beta_{m-j} \equiv \frac{m_j m_{j+2}}{(m_{j+1})^2} \quad (3.48)$$

From the above equation, a series of α_{m-j} and β_{m-j} can be obtained, i.e. $\alpha_{m-1}, \alpha_{m-2}, \alpha_{m-3}, \dots$ and $\beta_{m-1}, \beta_{m-2}, \beta_{m-3}, \dots$. It can be found that the series of α_{m-j} does not converge and the values of β_{m-j} fluctuate. By representing the values of β_{m-j} in the Cartesian coordinates with j being the horizontal axis and β_{m-j} being the vertical axis, the fluctuation of β_{m-j} resembles a wave shape and is called the β wave. The two approximate dominant poles can be found from this waveform. In the same way, the α wave can also be obtained.

3.5.4 Two-Pole Approximation

The two-pole approximate transfer function can be expressed in partial-fraction form as

$$\hat{H}(s)_{2p} = \frac{\hat{k}_1}{s - \hat{p}_1} + \frac{\hat{k}_2}{s - \hat{p}_2} \quad (3.49)$$

where $\hat{p}_1, \hat{p}_2$ are the two approximate dominant poles and $\hat{k}_1, \hat{k}_2$ are the corresponding approximate residues. When the system is underdamped, the two poles and two residues are complex conjugate pairs respectively.

For the circuit considered (in numerical example part Section 3.7.2), the loads are capacitive. Therefore, the DC gains are unity. Furthermore, since $\hat{h}(t)|_{t=0} = 0$, we have

Chapter 3: Computation of Approximate Delay

$$\hat{k}_1 + \hat{k}_2 = 0 \quad (3.50)$$

$$-\frac{\hat{k}_1}{\hat{p}_1} - \frac{\hat{k}_2}{\hat{p}_2} = 1 \quad (3.51)$$

Based on eqns. (3.50) and (3.51), eqn. (3.49) can be written as

$$\hat{H}(s)_{2p} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad (3.52)$$

where ω_n is the natural undamped frequency and ζ is the damping ratio (Kuo 1995):

$$\omega_n = \sqrt{\hat{p}_1 \hat{p}_2} \quad \text{and} \quad \zeta = -\frac{\hat{p}_1 + \hat{p}_2}{2\sqrt{\hat{p}_1 \hat{p}_2}} \quad (3.53)$$

The moments of interconnects can also be obtained by the poles and residues. For a system with N poles (N^{th} -order system), the j^{th} moment satisfies the following relation with the poles and residues:

$$m_j = -\sum_{i=1}^N \frac{k_i}{p_i^{j+1}} \quad (3.54)$$

where p_i and k_i are the i^{th} pole and its corresponding residue respectively. Hence, the ratio of two successive moments can be expressed as

$$\frac{m_j}{m_{j+1}} = p_1 \cdot \frac{1 + \frac{k_2}{k_1} \left(\frac{p_1}{p_2}\right)^{j+1} + \frac{k_3}{k_1} \left(\frac{p_1}{p_3}\right)^{j+1} + \dots + \frac{k_n}{k_1} \left(\frac{p_1}{p_N}\right)^{j+1}}{1 + \frac{k_2}{k_1} \left(\frac{p_1}{p_2}\right)^{j+2} + \frac{k_3}{k_1} \left(\frac{p_1}{p_3}\right)^{j+2} + \dots + \frac{k_n}{k_1} \left(\frac{p_1}{p_N}\right)^{j+2}} \quad (3.55)$$

Let $\hat{p}_1$ and $\hat{p}_2$ be the two approximate dominant poles and assume that the other poles are of much larger magnitude. Represent the poles using polar coordinates:

$$\hat{p}_1 = r_p e^{i\theta} \quad \text{and} \quad \hat{p}_2 = r_p e^{-i\theta} \quad (3.56)$$

Considering eqns. (3.48) and (3.50), eqn. (3.55) becomes

$$\alpha_{m-j} \equiv \frac{m_j}{m_{j+1}} \approx \hat{p}_1 \frac{1 - e^{i(2j+2)\theta}}{1 - e^{i(2j+4)\theta}} = r_p \frac{\sin((j+1)\theta)}{\sin((j+2)\theta)} \quad (3.57)$$

Eliminating r_p using m_j/m_{j+1} and m_{j+1}/m_{j+2} , we have

$$\beta_{m_j} \equiv \frac{m_j m_{j+2}}{(m_{j+1})^2} \approx \frac{\sin((j+1)\theta) \sin((j+3)\theta)}{(\sin((j+2)\theta))^2} \quad (3.58)$$

Taking the first crest (Shao and Chen 2000) in the β wave from $j=1$, eqn. (3.58) gives the corresponding solution set of θ . Then, θ is solved as the largest one in the solution set and r_p is obtained from eqn. (3.57). Substituting θ and r_p into eqn. (3.56), the approximate dominant poles can be computed.

The approximate transfer function then can be computed by substituting these two poles into eqns. (3.52) and (3.53).

3.5.5 Algorithm of Two-Pole Method

The two-pole method to compute the new approximate transfer function can be summarized in the following algorithm.

Algorithm 3.3:

Step (0): Initialization: (1) Use the equivalent circuits to replace the transmission lines. (2) For the obtained equivalent network, use the state-space method in Section 2.3.1 to compute the state parameters $\mathbf{A}$, $\mathbf{b}$ and $\mathbf{c}$.

Step (1): Compute the moments of the circuit:

$$m_i = \mathbf{c}^T \mathbf{v}_i, \quad (0 \leq i < 2q) \quad \text{where} \quad \mathbf{v}_i = -\mathbf{A}^{-i-1} \mathbf{b}.$$

Step (2): Compute $\alpha_{m_j} \equiv \frac{m_j}{m_{j+1}}$ and $\beta_{m_j} \equiv \frac{m_j m_{j+2}}{(m_{j+1})^2}$.

Then build the α wave and β wave shown as the fluctuations of α_{m_j} and β_{m_j} , whose values are illustrated in the Cartesian coordinates with j being the horizontal axis and α_{m_j} and β_{m_j} , being the vertical axis, respectively.

Step (3): In the β wave, take the first crest denoted as β_{crest_1} where $\beta_{crest_1} = \beta_{m_j_1}$.

After that, obtain the corresponding point, $\alpha_{m_j_1}$, in the α wave.

Step (4): Setting $j = j_1$, compute θ through solving,

$$\beta_{m-j} \equiv \frac{m_j m_{j+2}}{(m_{j+1})^2} \approx \frac{\sin((j+1)\theta) \sin((j+3)\theta)}{(\sin((j+2)\theta))^2}.$$

Take the largest θ in the solution set as the correct one then substitute θ into the below equation and compute r_p ,

$$\alpha_{m-j} \equiv \frac{m_j}{m_{j+1}} \approx \hat{p}_1 \frac{1 - e^{i(2j+2)\theta}}{1 - e^{i(2j+4)\theta}} = r_p \frac{\sin((j+1)\theta)}{\sin((j+2)\theta)}.$$

Step (5): Compute two approximate poles from

$$\hat{p}_1 = r_p e^{i\theta} \quad \text{and} \quad \hat{p}_2 = r_p e^{-i\theta}.$$

Then, the corresponding approximate residues, $\hat{k}_1$ and $\hat{k}_2$, can be obtained from eqn. (3.26).

Step (6): Obtain the approximate transfer function in s -domain from eqn. (3.49)

$$\hat{H}(s)_{2p} = \frac{\hat{k}_1}{s - \hat{p}_1} + \frac{\hat{k}_2}{s - \hat{p}_2}. \text{ Exit.}$$

3.6 Approximate Delay Computation

In the above sections, three approaches (the AWE technique, the Padé approximation and the two-pole method) are used to compute the approximate network transfer functions.

After obtaining the approximate transfer function, $\hat{H}(s)$, the approximate unit-step response, $\hat{Y}_R(s)$, can be computed as

$$\hat{Y}_R(s) = \frac{1}{s} \cdot \hat{H}(s) \tag{3.60}$$

The approximate unit-step response in t -domain, $\hat{y}_R(t)$, can be computed as described in Section 2.2.3. Thus, the approximate delay of the network can be evaluated as mentioned in Section 2.2.4.

3.7 Numerical Examples and Analysis

3.7.1 Delay Evaluation Using AWE Technique and Padé Approximation

Three types of circuits having the same structure shown in Figure 2.2, namely, Butterworth low-pass filter, Chebyshev low-pass filter and Bessel filter are chosen as examples. The AWE and Padé methods are used to compute the approximate network transfer functions, and the wave-chain method (Section 2.3.2) is used to compute the exact transfer functions. The approximate delays from different methods are compared.

(A) Butterworth Low-Pass Filter:

Butterworth low-pass filters with increasing order number N are chosen as the circuits for illustration. They have the ladder network structure shown in Figure 2.2 and the values of their elements are given by (Lam 1979):

$$\begin{cases} R_s = 1\Omega, R_l = 1\Omega, \\ C_j = 2 \sin[(2j-1)\pi / 2N], \quad j = \text{odd}, \\ L_j = 2 \sin[(2j-1)\pi / 2N], \quad j = \text{even}. \end{cases} \quad (3.61)$$

The values of exact and approximate delays are listed in Tables 3.3-3.5. In these tables, the “*error*” is defined as the absolute value of the difference between an approximate delay, $\hat{T}_D$, and an exact one, T_D , i.e.,

$$\text{error} = |\hat{T}_D - T_D| \quad (3.62)$$

From Tables 3.3, 3.4 and 3.5, it could be observed that with increasing order, the accuracies of both approximate methods decrease gradually. Moreover, it could be seen that the exact method uses the most computation time and the AWE technique is the most efficient with good accuracy. The computation time of Padé approximation is more than that of the AWE technique, but is still much less than the computation time of the wave-chain method. When the order of network is low, the approximate delays from both methods are very close to the exact one and their accuracies are almost the same. Even though with high network order, the Padé approximation gradually shows a slightly better performance than the AWE technique, the AWE technique can be regarded as the best

approximation method because of its short computation time.

Table 3.3 Exact delays of Butterworth low-pass filters

Network Order, N (Actual pole number)	Exact Delay, T_D (s)	Computation Time (s)
3	2.13511165346964	1.302
5	3.49604898629000	4.757
8	5.49371910731426	10.075
10	6.81167589004000	15.913
12	8.12315116913232	24.154
15	10.08212460197321	36.135
18	12.03419663768242	48.125
20	13.33275483251534	55.803
25	16.57188944157024	85.439
30	19.80339558289302	117.720
35	23.02955150212134	244.939
40	26.24746784440618	337.429
45	29.46594669559031	446.572
50	32.68868940305925	755.736
55	35.90016705448344	1751.807
60	39.11144125192838	2318.751

Table 3.4 Approximate delays of Butterworth low-pass filters using the AWE technique

Network Order, N (Actual pole number)	Used Pole Number	Approximate Delay, $\hat{T}_D$ (s)	Computation Time (s)	<i>error</i> (s)
3	3	2.13511165346959	0.410	4.1000e-13
5	5	3.49604898628822	1.237	1.7800e-12
8	8	5.49371910732213	2.473	7.8700e-12
10	10	6.81167589066600	4.271	6.2600e-10
12	10	8.13315204203422	4.362	0.0100
15	10	10.09819511001399	5.012	0.0161
18	10	12.18215263493820	5.492	0.1480
20	10	13.46419975809092	5.863	0.1314
25	10	16.52356282032016	6.390	0.0483
30	10	19.56945076279849	6.702	0.2339
35	10	22.63906202307632	6.943	0.3905
40	10	25.72920576323572	7.323	0.5183
45	10	28.83431024907968	7.963	0.6316
50	10	31.95015736961476	8.144	0.7385
55	10	35.07389167805270	8.459	0.8263
60	10	38.20350379170833	8.611	0.9079

Chapter 3: Computation of Approximate Delay

Table 3.5 Approximate delays of Butterworth low-pass filters using Padé approximation

Network Order, N (Actual pole number)	M and J (The max of M is 10)	Approximate Delay, $\hat{T}_D$ (s)	Computation Time (s)	error (s)
3	M=3 and J=3	2.13511165346960	1.773	4.0000e-13
5	M=5 and J=4	3.49604898628821	5.708	1.7900e-12
8	M=8 and J=8	5.49371910733982	15.592	9.8200e-12
10	M=10 and J=9	6.81167589062017	19.750	5.8017e-10
12	M=10 and J=9	8.13315204203422	24.497	0.0100
15	M=10 and J=9	10.09819511001400	29.680	0.0161
18	M=8 and J=7	12.02843443523007	38.806	0.0058
20	M=7 and J=7	13.33800669444765	43.820	0.0053
25	M=8 and J=8	16.59144477649761	59.085	0.0196
30	M=9 and J=9	19.73067320973980	72.734	0.0727
35	M=9 and J=9	22.79265701963302	95.177	0.2369
40	M=9 and J=9	25.88425415669828	107.815	0.3632
45	M=9 and J=9	28.99539831427025	123.998	0.4705
50	M=9 and J=9	32.11984544883428	148.293	0.5688
55	M=9 and J=9	35.25369105049818	170.985	0.1798
60	M=9 and J=9	38.39440502663312	200.739	0.7170

Figure 3.3 Comparison of step responses using AWE, Padé and wave-chain methods (Butterworth low-pass filter, $N = 15$)

Figure 3.4 Comparison of step responses using AWE, Padé and wave-chain methods (Butterworth low-pass filter, $N = 30$)

Also, the step responses based on the AWE and Padé methods were compared with that of the exact method. The 15th-order and 30th-order Butterworth low-pass filters were chosen as examples. The results are shown in Figures 3.3 and 3.4. Here, the pole number used in the AWE technique is 10, which is the same as that in Table 3.4.

From Figures 3.3 and 3.4, it is obvious that the accuracies of both approximate methods decrease gradually when the network order increases. Also, the two approximate methods have close accuracies when the network order is not very high.

(B) Chebyshev Low-Pass Filter:

A typical circuit structure to realize a Chebyshev filter is given in Figure 2.2. An N^{th} -order Chebyshev Low-pass filter (Lam 1979) has the following parameters:

$$R_s = 1\Omega \quad \text{and} \quad \varepsilon = \sqrt{10^{(A_{\max}/10)} - 1} \quad (3.63)$$

where $A_{\max}$ = the maximum passband ripple attenuation (dB), ε = the ripple parameter

Chapter 3: Computation of Approximate Delay

(dB). $A_{\max}$ is arbitrary. By defining

$$\alpha = \frac{4R_l}{(R_l + 1)^2} \quad (\text{when } N \text{ is odd}) \quad (3.64)$$

and

$$\alpha = \frac{4R_l}{(R_l + 1)^2} [1 + \varepsilon^2] \leq 1 \quad (\text{when } N \text{ is even}) \quad (3.65)$$

where R_l is arbitrary, except when N is even, R_l has to satisfy the inequality constraint of eqn. (3.65), and by letting

$$\begin{cases} \alpha_i = 2 \sin \frac{\pi i}{2N}, \quad \beta_i = 2 \cos \frac{\pi i}{2N} \\ \gamma = \left[\frac{1}{\varepsilon} + \sqrt{\frac{1}{\varepsilon^2} + 1} \right]^{1/N}, \quad \iota = \left[\sqrt{\frac{1-a}{\varepsilon^2}} + \sqrt{\frac{1-a}{\varepsilon^2} + 1} \right]^{1/N} \\ x = \gamma - \frac{1}{\gamma}, \quad y = \iota - \frac{1}{\iota} \end{cases} \quad (3.66)$$

the element values of Chebyshev low-pass filter can be found by a recursive relationship:

$$\begin{cases} C_{2m-1} L_{2m} = \frac{4\alpha_{4m-3} \alpha_{4m-1}}{b_{2m-1}(x, y)} \\ C_{2m+1} L_{2m} = \frac{4\alpha_{4m-1} \alpha_{4m+1}}{b_{2m}(x, y)} \end{cases} \quad (3.67)$$

$$\text{with} \quad C_1 = 2\alpha / (x - y_1) \quad (3.68)$$

where the function $b_i(x, y)$ is defined by

$$b_i(x, y) = x^2 - \beta_{2i}xy + y^2 + \alpha_{2i}^2 \quad (3.69)$$

and the value of m is determined by N :

$$\begin{cases} m = 1, 2, \dots, (N-1)/2 & (\text{when } N \text{ is odd}) \\ m = 1, 2, \dots, N/2 & (\text{when } N \text{ is even}) \end{cases} \quad (3.70)$$

Given the parameters N and $A_{\max}$, we can calculate the values of all the dependent variables in eqn. (3.66). By eqn. (3.68), C_1 is found. With C_1 known and $m=1$, eqn. (3.67) is used to find L_2 and C_3 . Then set $m=2$, and repeat the process of eqn. (3.67). This process can be repeated as m is incremented.

Chapter 3: Computation of Approximate Delay

Taking Chebyshev low-pass filters as example circuits, their exact, AWE and Padé delays can be obtained. All results, which are listed in Tables 3.6-3.8 and Figures 3.5-3.6, show similar trend as for Butterworth low-pass filters.

Table 3.6 Exact delays of Chebyshev low-pass filters with $A_{\max} = 0.1$ dB

Network Order, N (Actual pole number)	R_l	Exact Delay, T_D (s)	Computation Time (s)
3	1	1.70976466315569	3.565
5	1	3.75318591693903	4.146
6	0.5	4.77263355953026	5.207
7	1	5.82563874924245	6.970
8	0.5	6.84067880085637	8.092
9	1	7.89227072266412	10.124
10	0.5	8.90123799156681	12.929
12	0.5	10.95445929706349	20.139
15	1	14.04820277942229	29.493
18	0.5	17.08270666456371	49.812
20	0.5	19.11813035663981	80.038
25	1	24.22572067103745	107.152
30	0.5	29.26287685829298	163.745
35	1	34.35492434948829	190.351
40	0.5	39.37449126607638	272.613

Table 3.7 Approximate delays of Chebyshev low-pass filters with $A_{\max} = 0.1$ dB using the AWE technique

Network Order, N (Actual pole number)	R_l	Used Pole Number	Approximate Delay, $\hat{T}_D$ (s)	Computation Time (s)	<i>error</i> (s)
3	1	3	1.70976466315569	0.932	0
5	1	5	3.75318591693942	1.281	3.9000e-13
6	0.5	6	4.77263355953215	1.552	1.8900e-12
7	1	7	5.82563875098451	1.936	1.7421e-9
8	0.5	8	6.84067878881723	2.180	1.2039e-8
9	1	9	7.89234039088932	2.326	6.9668e-5
10	0.5	9	8.86716095444942	2.333	0.0341
12	0.5	9	10.85914035211446	2.337	0.0953
15	1	9	13.97204731889767	2.472	0.0762
18	0.5	9	16.80223029095223	2.573	0.2805
20	0.5	9	18.78715367901792	2.661	0.3310
25	1	9	23.97782808872411	3.318	0.2479
30	0.5	9	28.74712691910704	3.494	0.5157
35	1	9	34.00561055791182	3.712	0.3493
40	0.5	9	38.74489962879751	3.900	0.6296

Table 3.8 Approximate delays of Chebyshev low-pass filters with $A_{\max} = 0.1\text{dB}$ using Padé approximation (The distribution of R_l is as same as that in Table 3.7)

Network Order, N (Actual pole number)	M and J (The max of M is 9)	Approximate Delay, $\hat{T}_D$ (s)	Computation Time (s)	<i>error</i> (s)
3	M=3 and J=3	1.70976466315568	2.023	1.0000e-14
5	M=5 and J=4	3.75318591693897	2.257	6.0000e-14
6	M=6 and J=5	4.77263355953521	2.374	4.9500e-12
7	M=7 and J=6	5.82563874796938	2.673	1.2731e-9
8	M=8 and J=7	6.84067879611849	3.185	4.7379e-9
9	M=9 and J=9	7.89242744666439	4.707	1.5672e-4
10	M=8 and J=7	8.86926242201205	6.624	0.0320
12	M=9 and J=9	10.87842763126402	8.230	0.0760
15	M=9 and J=8	13.96853654164051	11.828	0.0797
18	M=9 and J=9	16.83618807715801	15.432	0.2465
20	M=9 and J=9	18.82130022478020	16.013	0.2968
25	M=9 and J=8	23.99199975415327	21.451	0.2337
30	M=9 and J=9	28.80240258297986	34.686	0.4605
35	M=9 and J=9	33.99315642496524	42.513	0.3618
40	M=8 and J=8	38.81285942246731	50.787	0.5616

Figure 3.5 Comparison of step responses using AWE, Padé and wave-chain methods (Chebyshev low-pass filter, $N = 15$ and $A_{\max} = 0.1\text{dB}$)

Figure 3.6 Comparison of step responses using AWE, Padé and wave-chain methods (Chebyshev low-pass filter, $N = 30$ and $A_{\max} = 0.1\text{dB}$)

(C) Bessel Filter:

Bessel filter has $R_l = 1\,\Omega$, and its other element values are listed in Table 3.9 (Williams and Taylor 1995). The unit of capacitors is F and the unit of inductors is H. Here, $R_s = 1\,\Omega$ and the results are shown in Tables 3.10-3.12 and Figure 3.7.

Table 3.9 Element values of Bessel filter

Network Order	R_s	C_1	L_2	C_3	L_4	C_5	L_6	C_7	L_8	C_9	L_{10}
2	1.0000	0.5755	2.1478								
3	1.0000	0.3374	0.9705	2.2034							
4	1.0000	0.2334	0.6725	1.0815	2.2404						
5	1.0000	0.1743	0.5072	0.8040	1.1110	2.2582					
6	1.0000	0.1365	0.4002	0.6392	0.8538	1.1126	2.2645				
7	1.0000	0.1106	0.3259	0.5249	0.7020	0.8690	1.1052	2.2659			
8	1.0000	0.0919	0.2719	0.4409	0.5936	0.7303	0.8695	1.0956	2.2656		
9	1.0000	0.0780	0.2313	0.3770	0.5108	0.6306	0.7407	0.8639	1.0863	2.2649	
10	1.0000	0.0672	0.1998	0.3270	0.4454	0.5528	0.6493	0.7420	0.8561	1.0781	2.2641

Again the conclusion is similar to that for Butterworth and Chebyshev low-pass filters. Both approximate methods have very good accuracies when the network order is low and the AWE technique is the most efficient when N is large.

Table 3.10 Exact delays of Bessel filters with $R_s = 1\Omega$

Network Order, N (Actual pole number)	Exact Delay, T_D (s)	Computation time (s)
2	1.22549094546618	3.926
3	1.68072280545106	4.807
4	2.06936871907129	6.549
5	2.39976944254414	8.422
6	2.68582221484648	9.252
7	2.94029072756370	10.215
8	3.17205113696519	12.357
9	3.38662089309912	14.341

Table 3.11 Approximate delays of Bessel filters with $R_s = 1\Omega$ using the AWE technique

Network Order, N (Actual pole number)	Used Pole Number	Approximate Delay, $\hat{T}_D$ (s)	Computation Time (s)	<i>error</i> (s)
2	2	1.22549094546618	1.763	0
3	3	1.68072280545106	2.424	0
4	4	2.06936871907126	2.794	3.0000e-14
5	5	2.39976944254390	2.414	2.4000e-13
6	6	2.68582221484189	3.095	4.5900e-12
7	7	2.94029072757287	3.946	9.1700e-12
8	8	3.17205114478614	4.356	7.8210e-9
9	9	3.38662095145470	6.039	5.8356e-8

Table 3.12 Approximate delays of Bessel filters with $R_s = 1\Omega$ using Padé approximation

Network Order, N (Actual pole number)	M and J (The max of M is 10)	Approximate Delay, $\hat{T}_D$ (s)	Computation Time (s)	<i>error</i> (s)
2	M=2 and J=2	1.22549094546618	1.172	0
3	M=3 and J=3	1.68072280545106	2.684	0
4	M=4 and J=3	2.06936871907129	4.717	0
5	M=5 and J=5	2.39976944254403	6.900	1.1000e-13
6	M=6 and J=5	2.68582221484570	9.764	7.8000e-13
7	M=7 and J=7	2.94029072752080	14.741	4.2900e-11
8	M=8 and J=7	3.17205113328602	17.014	3.6792e-9
9	M=9 and J=8	3.38662109078980	24.655	1.9769e-7

Figure 3.7 Comparison of step responses using AWE, Padé and wave-chain methods (Bessel filter, $N = 8$ and $R_s = 1\Omega$)

3.7.2 Delay Evaluation Using Two-Pole Method

Considering the circuit shown in Figure 3.8, parameters for line 1 are $d_{trans} = 6\text{ cm}$, $R_{trans} = 3.06\ \Omega\text{cm}^{-1}$, $L_{trans} = 1.30\text{ Hcm}^{-1}$, $C_{trans} = 2.98 \times 10^{-3}\text{ Fcm}^{-1}$, and for line 2 are $d_{trans} = 5\text{ cm}$, $R_{trans} = 15.00\ \Omega\text{cm}^{-1}$, $L_{trans} = 3.33\text{ nHcm}^{-1}$, $C_{trans} = 1.15\text{ pFcm}^{-1}$. Using the equivalent “T” circuit to describe the transmission line, it can be transformed into an 8th-order equivalent circuit as shown in Figure 3.9.

Figure 3.8 Transmission line: $R_1 = 1.6\ \Omega$, $R_2 = 8.9\ \Omega$, $C_1 = 10\text{pF}$, $C_2 = 5\text{pF}$

Figure 3.9 The equivalent T-circuit

The values of elements in Figure 3.9 are computed as

$$R_{T_1} = 9.18 \, \Omega, L_{T_{11}} = L_{T_{12}} = 3.64886825 \text{H}, C_{T_1} = 0.01788000 \text{F}, R_{T_2} = 37.5 \, \Omega,$$

$$L_{T_{21}} = L_{T_{22}} = 6.97734375 \text{H} \text{ and } C_{T_2} = 0.00575000 \text{F}.$$

The exact delay can be obtained by the state-space method as

$$T_D = 0.67210336178693 \text{ s}$$

The moments of circuit, m_j , can be obtained from eqn. (3.4). Eqn. (3.48) gives the definition of α_{m_j} and β_{m_j} can be computed from the moments. All results are listed in Table 3.13. It can be seen that the series of α_{m_j} does not converge and the values of β_{m_j} fluctuate. After illustrating the values of β_{m_j} in the Cartesian coordinates with j being the horizontal axis and β_{m_j} being the vertical axis, the fluctuation of β_{m_j} resembles a wave shape. Thus, the β wave is obtained.

According to Table 3.13, the first crest in the β wave is found as β_{m_3} . It corresponds to a pair of approximate poles, $\hat{p}_1$ and $\hat{p}_2$. After getting the corresponding residues using eqn. (3.26), the two-pole approximate transfer function of the circuit is computed. The approximate delay, $\hat{T}_D$, is then obtained. Table 3.14 compares the approximate delay from the two-pole method to that from the AWE method with pole number $q=2$. Also, their execution times are compared. In Table 3.14, the “error” is

defined as in eqn. (3.62).

Table 3.13 Moments and ratios (crest value is underlined)

j	m_j	α_{m_j}	β_{m_j}
0	1.000000000000000	-1.74120049287858	0.36659362142877
1	-0.57431640071890	-4.74967482001564	-0.70373600395316
2	0.12091699379054	6.74922811016469	-7.83562410916777
3	0.01791567743998	-0.86135169529993	<u>0.27111398021412</u>
4	-0.02079949170326	-3.17708328659280	0.23340969468799
5	0.00654672535373	-13.61161665045555	-13.70648464891062
6	-0.00048096603966	0.99307860469806	-0.50840987319375
7	-0.00048431819736	-1.95330314586477	<u>0.40666033773584</u>
8	0.00024794830151	-4.80327921021304	-0.41277889366377
9	-0.00005162063054	11.63644576780499	-18.81617551518455
10	-0.00000443611663	-0.61842778615742	0.21527971941702
11	0.00000717321687	-2.87267090384603	<u>0.32898531833485</u>
12	-0.00000249705487	-8.73191216673723	-4.30827841778889
13	0.00000028596885	2.02677527308429	-1.26266635611764
14	0.00000014109549	-1.60515504611692	0.39496272925543
15	-0.00000008790147	-4.06406713145540	-0.03743073947803
16	0.00000002162894	108.5756570168885	-1268.664032657222
17	0.00000000019921	-0.08558267139447	0.03455806170946
18	-0.00000000232765	-2.47648933884030	0.38526963403236
19	0.00000000093990	-6.42793804671435	-1.63700594563836
20	-0.00000000014622		

Table 3.14 Approximate delays and execution time of two-pole method and AWE method
(Cf: Exact delay, $T_D = 0.67210336178693$ s)

Approximate Method	Approximate Pole Number	Approximate Delay $\hat{T}_D$ (s)	<i>error</i> (s)	Execution Time (s)
Two-Pole Method	2 (from β_{m_3})	0.40765454855683	0.2644	2.352
AWE method	2	0.47019670104191	0.2019	4.665

It is found from Table 3.14 that the AWE method using 2 poles is more accurate than the two-pole method in delay estimation. However, the two-pole method has a better execution time.

3.8 Summary

Although it is desirable to obtain the exact transfer function of a given circuit, it is generally impractical for a very high-order interconnect circuit because the computation time can be prohibitively large. Thus, some efficient algorithms are studied to approximate the original system behaviour with a lower order model. They are the asymptotic waveform evaluation (AWE) technique, the Padé approximation and the two-pole method.

The AWE technique approximates the Laplace-domain transfer function of a linear network by a reduced-order model, containing only a relatively small number of dominant poles and residues. The Padé approximation is also based on dominant poles and residues. However, it searches the Padé table to select the appropriate orders for the numerator and denominator polynomials. The two-pole method obtains two dominant approximate poles from the crest in the β wave and then produces the two-pole approximate transfer function. With the obtained transfer function, the corresponding approximate delay can then be computed.

Comparing the AWE technique with the Padé approximation, the AWE technique is more efficient and has good accuracy. The computation time of the Padé approximation is still much less than that of the exact method, such as the wave-chain method. When the order of network is low, the approximate delays from both methods are very close to the exact delay and their accuracies are almost the same. With increasing order, the accuracies of both approximate methods decrease but the Padé approximation shows a slightly better performance. The two-pole method is compared to the AWE technique. It is found that the AWE method using only two poles is more accurate than the two-pole method. However, the two-pole method is much more efficient.

In general, the AWE technique can be regarded as the best approximation method because of its short computation time and good accuracy.

Chapter 4: Improvement of Approximate Delay Accuracy by the Addition of Extra Poles

4.1 Introduction

In Chapter 3, it is known that the AWE technique is an accurate approximation method with superior efficiency. Despite its efficiency, the AWE suffers from a number of fundamental numerical limitations. In fact, the AWE is a specific type of a well-known Padé approximation (Chiprout and Nakhla 1994, Celik *et al.* 2002). A q^{th} -order standard AWE model is a Padé approximation of type $[(q-1)/q]$. As such, the AWE inherits the instability of Padé approximation. In particular, each run of the AWE method produces only a few accurate poles. Along with accurate poles, some unstable approximate poles in the right-half s -plane (RHP) will be generated and should be discarded.

Available remedial techniques such as scaling, frequency shifting and complex frequency hopping (Chiprout and Nakhla 1994, Celik *et al.* 2002) are sometimes heuristic, hard to apply automatically, and may be computationally expensive. More recently, the Padé via Lanczos (PVL) process (Feldmann and Freund 1995) has been hailed as a numerically more stable algorithm than the AWE approximation. However, the PVL approach cannot provide an unsupervised measure of the error for a given order of Padé approximation and, therefore, the termination of the Lanczos procedure is largely heuristic. In this chapter, the interpolation method is applied to improve the AWE accuracy loss caused by the discarded approximate poles in the RHP. Based on the useable approximate poles from the AWE, the new approximate poles in the left-half s -plane (LHP) can be obtained by using the interpolation method. By adding these poles into the original useable pole set, the total number of stable poles is increased and the accuracy of new approximate delay is improved.

Chapter 4: Improvement of Approximate Delay Accuracy by the Addition of Extra Poles

Two-pole method is also presented in Chapter 3 to compute approximate system delays using two-pole approximations. Although it is much more efficient than other approximation methods, two-pole method has same obvious shortage: low accuracy. To enhance the accuracy of delay evaluation using the above two-pole method, a new technique called the extended two-pole method is proposed in this chapter by using additional pairs of approximate dominant poles.

4.2 Improved AWE-Based Delay by Pole Interpolation

4.2.1 Generation of New Stable Poles Using Interpolation

Let $\hat{p}_i = \alpha_i + j\beta_i$, $i = 1, 2, \dots, q$, designate the q poles computed from the AWE, where α_i and β_i are the respective real and imaginary parts. The AWE pole set is denoted as

$$[pole]_{AWE} = [\hat{p}_1, \hat{p}_2, \dots, \hat{p}_q] \quad (4.1)$$

For a chosen number of poles, q , the AWE may produce r number of poles in the right-half s -plane (RHP); Assume that these pole set in the RHP are denoted as

$$[pole]_{RHP} = [\hat{p}_1, \hat{p}_2, \dots, \hat{p}_r] \quad (4.2)$$

These poles are unstable and will not be used since the system response which will contain terms of the form $e^{\hat{p}_i t}$ will increase without bound if $\hat{p}_i$ have positive α_i . Thus, the number of remaining poles which can be used for the delay calculation is $v = q - r$ and the poles are denoted as

$$[pole]_{useable} = [\hat{p}_{r+1}, \hat{p}_{r+2}, \dots, \hat{p}_q] \quad (4.3)$$

Here, we attempt to generate a number of stable poles, g ($g \leq r$), from interpolation on the existing v useable poles so that the total number of stable poles is increased to $v + g$ (with $(v + g) \leq q$) in order to improve the accuracy of delay computation.

Chapter 4: Improvement of Approximate Delay Accuracy by the Addition of Extra Poles

The following interpolation approach is investigated: an interpolation is performed on the ν acceptable poles for computing their real and imaginary values. Then the old imaginary values of those r discarded poles are used to obtain the corresponding new real values in the left-half s -plane (LHP) from the interpolated function.

Thus, new acceptable poles are generated and consist of the old imaginary values as their imaginary parts as well as newly obtained real values as their real parts. It is known that the newly obtained poles are the restored poles from original unstable poles in the RHP. The restored approximate poles from pole interpolation are described as

$$\hat{p}_{restored_i} = \alpha_{restored_i} + j\beta_{restored_i}, \quad i = 1, 2, \dots, g. \quad (4.4)$$

The restored pole collection from pole interpolation is represented as

$$[pole]_{restored} = [\hat{p}_{restored_1}, \hat{p}_{restored_2}, \dots, \hat{p}_{restored_g}] \quad (4.5)$$

Furthermore, in the process of pole interpolation, the following conditions are enforced:

- (i) After applying pole interpolation, new poles should be chosen from the well-behaved section of the interpolated function. The well-behaved section of the interpolated function is defined as the area not too close to two original poles used to generate new restored poles (Fausett 1999).
- (ii) From the formulation of AWE, Padé approximation may generate extremely small “parasitic” poles (close to the origin) (Chiprout and Nakhla 1994) with very small residues. The moduli of these poles are smaller than the moduli of the actual poles nearest the origin. These parasitic poles may be stable but they are the result of numerical errors in the moments and are thus removed before interpolation and will not be restored.
- (iii) Some of those r unstable poles (not close to the origin) may also have very small residues. It is discovered through interpolation that they are actually repeated poles because their restored real and original imaginary values are numerically very close to some of the existing acceptable poles, and are also removed.
- (iv) All the new residues are recomputed using the same moments. Interpolated poles are acceptable only if their corresponding residues are not very small.

Chapter 4: Improvement of Approximate Delay Accuracy by the Addition of Extra Poles

4.2.2 Evaluation of New Approximate Delay

Using restored poles to replace the unstable AWE poles or only adding the new ones to the original useable AWE pole set, the new pole set after remedy by restored poles (RRP) can be constructed as

$$[pole]_{RRP} = [\hat{p}_{r+1}, \hat{p}_{r+2}, \dots, \hat{p}_q, \hat{p}_{restored_1}, \hat{p}_{restored_2}, \dots, \hat{p}_{restored_g}] \quad (4.6)$$

Here, g restored poles are used for remedy and the useable approximate pole number increases to $v + g$. Corresponding to the new approximate pole set shown in eqn. (4.6), the corresponding residue set after RRP can be computed from eqn. (3.26) (in Chapter 3) and denoted as

$$[residue]_{RRP} = [\hat{k}_{r+1}, \hat{k}_{r+2}, \dots, \hat{k}_q, \hat{k}_{restored_1}, \hat{k}_{restored_2}, \dots, \hat{k}_{restored_g}] \quad (4.7)$$

Then from eqn. (3.23), the new approximate transfer function in s -domain can be obtained from

$$\hat{H}_{new}(s) = \sum_{i=r+1}^q \frac{\hat{k}_i}{s - \hat{p}_i} + \sum_{j=1}^g \frac{\hat{k}_{restored_j}}{s - \hat{p}_{restored_j}} \quad (4.8)$$

The corresponding unit-step response is given by $\hat{Y}_{R_new}(s) = s^{-1} \hat{H}_{new}(s)$, i.e.,

$$\begin{aligned} \hat{Y}_{R_new}(s) &= \sum_{i=r+1}^q \frac{\hat{k}_i}{s(s - \hat{p}_i)} + \sum_{i=1}^g \frac{\hat{k}_{restored_i}}{s(s - \hat{p}_{restored_i})} \\ &= \sum_{i=r+1}^q \frac{\hat{k}_i}{\hat{p}_i} \left(\frac{1}{s - \hat{p}_i} - \frac{1}{s} \right) + \sum_{i=1}^g \frac{\hat{k}_{restored_i}}{\hat{p}_{restored_i}} \left(\frac{1}{s - \hat{p}_{restored_i}} - \frac{1}{s} \right) \end{aligned} \quad (4.9)$$

with $v = q - r$ and $(v + g) \leq q$.

Applying inverse Laplace transform, the corresponding time-domain step response is given by

$$\hat{y}_{R_new}(t) = \sum_{i=r+1}^q \frac{\hat{k}_i}{\hat{p}_i} (e^{\hat{p}_i t} - 1)u(t) + \sum_{i=1}^g \frac{\hat{k}_{restored_i}}{\hat{p}_{restored_i}} (e^{\hat{p}_{restored_i} t} - 1)u(t) \quad (4.10)$$

Thus, the new approximate delay can be obtained by using the method mentioned in Section 2.2.5.

4.2.3 Algorithm of Improved AWE-Based Delay by Pole Interpolation

The process to improve AWE-based delay by pole interpolation can be summarized in the following algorithm.

Algorithm 4.1:

Step (0): Initialize: obtain enough moments of the network and compute the original approximate pole set $[pole]_{AWE}$ using the AWE method.

Step (1): Delete the unstable poles $[pole]_{RHP}$ and the parasitic poles from $[pole]_{AWE}$, and get the useable approximate pole set $[pole]_{useable}$.

Step (2): For computing a restored pole $\hat{p}_{restored_i}$ by pole interpolation, a discarded pole $\hat{p}_z$ in $[pole]_{RHP}$ ($1 \leq z \leq r$), whose imaginary part is between the well-behaved section of imaginary parts of useable poles, is chosen. Use the imaginary part of $\hat{p}_z$ as that of $\hat{p}_{restored_i}$, i.e.,

$$\hat{p}_{restored_i} = \alpha_{restored_i} + j\beta_{restored_i} = \alpha_{restored_i} + j\beta_z.$$

Step (3): Choose a type of interpolation method such as Lagrange interpolation, Cubic-spline interpolation or Hermite interpolation (Fausett 1999). Then, use the selected interpolation function to compute the new imaginary part $\alpha_{restored_i}$ from the interpolation of $[pole]_{useable}$.

Step (4): Go to Step (3) until obtaining enough $\hat{p}_{restored_i}$ to construct $[pole]_{restored}$.

Step (5): Compute the new pole collection after remedy by restored poles (RRP)

$$[pole]_{RRP} = [pole]_{useable} + [pole]_{restored}.$$

Compute the corresponding residue collection $[residue]_{RRP}$ from eqn. (3.26).

Step (6): Compute the new transfer function in s -domain from

$$\hat{H}_{new}(s) = \sum_{i=r+1}^q \frac{\hat{k}_i}{s - \hat{p}_i} + \sum_{j=1}^g \frac{\hat{k}_{restored_j}}{s - \hat{p}_{restored_j}}.$$

Step (7): Compute the step response in t -domain from

Chapter 4: Improvement of Approximate Delay Accuracy by the Addition of Extra Poles

$$\hat{y}_{R_new}(t) = \sum_{i=r+1}^q \frac{\hat{k}_i}{\hat{p}_i} (e^{\hat{p}_i t} - 1)u(t) + \sum_{i=1}^g \frac{\hat{k}_{restored_i}}{\hat{p}_{restored_i}} (e^{\hat{p}_{restored_i} t} - 1)u(t).$$

Step (8): Following Step (6) to Step (10) of Algorithm 2.1 in Section 2.2.5, the delay can be obtained. Exit.

In Step (3), Cubic-spline interpolation and Hermite interpolation have the corresponding built-in functions `terpl(x,y,xnew, 'spline')` and `terpl(x,y,xnew, 'pchip')` (Etter 1997, Fausett 1999) in MATLAB v.6.5 (Matlab 1996, Fausett 1999).

4.3 Delay Estimation by Extended Two-Pole Method

In this section, the two-pole method described in Section 3.5 is extended to produce more approximate poles and zeros for system approximation. Also, the interconnect networks including transmission lines are mainly considered to evaluate the operation of two-pole method extension.

4.3.1 Generation of α Wave and β Wave

For an interconnect network including transmission lines, Section 3.5.2 has represented the method to calculate its moments, m_i ($i \geq 0$). Let α_m represent the ratio of successive moments and β_m represent the ratio of successive α_m 's:

$$\alpha_{m_j} \equiv \frac{m_j}{m_{j+1}} \quad \beta_{m_j} \equiv \frac{m_j m_{j+2}}{(m_{j+1})^2} \quad (4.11)$$

Then, α wave and β wave can be established as the fluctuations of α_{m_j} and β_{m_j} , whose values are illustrated in the Cartesian coordinates with j being the horizontal axis and α_{m_j} and β_{m_j} , being the vertical axis, respectively. Details about generation of α wave and β wave have been stated in Section 3.5.3.

4.3.2 Extension of Two-Pole Approximation

In Section 3.5.4, two-pole method gives two dominant poles as

$$\hat{p}_1 = r_p e^{i\theta} \quad \text{and} \quad \hat{p}_2 = r_p e^{-i\theta} \quad (4.12)$$

The corresponding θ can be solved by substituting the different values of β_{m-j} shown in the different β – wave crest into the following equation:

$$\beta_{m-j} \equiv \frac{m_j m_{j+2}}{(m_{j+1})^2} \approx \frac{\sin((j+1)\theta) \sin((j+3)\theta)}{(\sin((j+2)\theta))^2} \quad (4.13)$$

The largest θ is taken from the solution set as the correct one. After that, the magnitude r_p can be found from

$$\alpha_{m-j} \equiv \frac{m_j}{m_{j+1}} \approx \hat{p}_1 \frac{1 - e^{i(2j+2)\theta}}{1 - e^{i(2j+4)\theta}} = r_p \frac{\sin((j+1)\theta)}{\sin((j+2)\theta)} \quad (4.14)$$

Eqn. (4.13) and (4.14) are same as eqn. (3.58) and (3.57) and repeated for convenience.

Since every crest of the β wave stands for a pair of approximate poles, more approximate poles can be obtained. If n' crests are chosen, we can get n' pairs of approximate poles, $\hat{p}_1, \hat{p}_2, \hat{p}_3, \hat{p}_4, \dots, \hat{p}_{2n'-1}, \hat{p}_{2n'}$, from eqns. (3.57) and (3.58). Using the same approach in Section 3.2.3, all corresponding residues, $\hat{k}_1, \hat{k}_2, \hat{k}_3, \hat{k}_4, \dots, \hat{k}_{2n'-1}, \hat{k}_{2n'}$, can be computed from eqn. (3.26). As a result, the approximate transfer function can be computed as

$$\hat{H}(s) = \sum_{j=1}^{2n'} \frac{\hat{k}_j}{s - \hat{p}_j} \quad (4.15)$$

4.3.3 Approximate Delay Computation

After obtaining the approximate transfer function, $\hat{H}(s)$, the approximate unit-step response, $\hat{Y}_R(s)$, can be computed as

Chapter 4: Improvement of Approximate Delay Accuracy by the Addition of Extra Poles

$$\hat{Y}_R(s) = \frac{1}{s} \cdot \hat{H}(s) \quad (4.16)$$

The approximate unit-step response in t -domain, $\hat{y}_R(t)$, can be computed as described in Section 2.2.3. Thus, the approximate delay of the network can be evaluated as mentioned in Section 2.2.4.

4.3.4 Algorithm of Delay Estimation by Extended Two-Pole Method

The extended two-pole method to compute the new approximate transfer function can be summarized in the following algorithm.

Algorithm 4.2:

Step (0): Initialize: use the equivalent circuits to replace the transmission lines. From the obtained equivalent network, use the state-space method in Section 2.3.1 to compute the state parameters $\mathbf{A}$, $\mathbf{b}$ and $\mathbf{c}$.

Step (1): Compute the moments of the circuit:

$$m_i = \mathbf{c}^T \mathbf{v}_i, \quad (0 \leq i < 2q) \quad \text{where} \quad \mathbf{v}_i = -\mathbf{A}^{-i-1} \mathbf{b}.$$

Step (2): Compute $\alpha_{m_j} \equiv \frac{m_j}{m_{j+1}}$ and $\beta_{m_j} \equiv \frac{m_j m_{j+2}}{(m_{j+1})^2}$.

Then build the α wave and β wave shown as the fluctuations of α_{m_j} and β_{m_j} , whose values are illustrated in the Cartesian coordinates with j being the horizontal axis and α_{m_j} and β_{m_j} , being the vertical axis, respectively.

Step (3): In the β wave, choose n' crests denoted as β_{crest_i} . Then, in the α wave, obtain the n' corresponding points denoted as α_{crest_i} .

Step (4): For every selected crest having $\beta_{crest_i} = \beta_{m_j}$, compute θ through solving,

$$\beta_{m_j} \equiv \frac{m_j m_{j+2}}{(m_{j+1})^2} \approx \frac{\sin((j+1)\theta) \sin((j+3)\theta)}{(\sin((j+2)\theta))^2}.$$

Chapter 4: Improvement of Approximate Delay Accuracy by the Addition of Extra Poles

Then substitute θ into the below equation and compute r_p ,

$$\alpha_{m-j} \equiv \frac{m_j}{m_{j+1}} \approx \hat{p}_1 \frac{1 - e^{i(2j+2)\theta}}{1 - e^{i(2j+4)\theta}} = r_p \frac{\sin((j+1)\theta)}{\sin((j+2)\theta)}.$$

Step (5): For every set of θ and r_p , compute two approximate poles

$$\hat{p}_1 = r_p e^{i\theta} \quad \text{and} \quad \hat{p}_2 = r_p e^{-i\theta}.$$

For n' selected crests, n' sets of θ and r_p lead to n' pairs of approximate poles, $\hat{p}_1, \hat{p}_2, \dots, \hat{p}_{2n'-1}, \hat{p}_{2n'}$. The corresponding approximate residues $\hat{k}_1, \hat{k}_2, \dots, \hat{k}_{2n'-1}, \hat{k}_{2n'}$ can then be computed from eqn. (3.26).

Step (6): Compute the approximate transfer function in s -domain, $\hat{H}(s) = \sum_{j=1}^{2n'} \frac{\hat{k}_j}{s - \hat{p}_j}$.

Step (7): Compute the unit-step response in s -domain, $\hat{Y}_R(s) = \hat{H}(s)/s$.

Step (8): Following Step (3) to Step (10) of Algorithm 2.1 in Section 2.2.5, the approximate delay can be obtained. Exit.

4.4 Numerical Examples and Analysis

4.4.1 Improved AWE-Based Delay by Pole Interpolation

For the AWE model, the exact transfer function of an interconnect network can be easily replaced by its approximate form. The delay, T_D , of a circuit is defined as the time at which the transient response would reach 50% of its final value (Pillage and Rohrer 1990, Chiprout and Nakhla 1994, Celik *et al.* 2002). The exact and approximate delays for many types of interconnect circuits have been computed and compared using conventional exact method (State-Space method) and the AWE, respectively. Improved results were always obtained for the AWE with interpolated poles where applicable. The following three examples are used for illustration. In this section, the “error” is defined as the absolute value of the difference between an approximate delay, $\hat{T}_D$, and an exact one, T_D , i.e.,

Chapter 4: Improvement of Approximate Delay Accuracy by the Addition of Extra Poles

$$error = |\hat{T}_D - T_D| \tag{4.17}$$

and the corresponding error rate φ is defined by

$$\varphi = error/T_D \tag{4.18}$$

(A) Chebyshev Low-Pass Filter

Consider an $N = 42$ Chebyshev low-pass filter (Lam 1979) whose properties are represented in Section 3.7.1 with the maximum passband ripple attenuation, $A_{\max} = 0.1$ dB, and its exact delay is found to be

$$T_D = 41.39427213s \tag{4.19}$$

Approximated by a $q = 10$ reduced AWE model, the AWE approximate poles and corresponding residues are computed and shown in Table 4.1. Notice that there are two unstable poles (underlined) in the RHP, i.e. $r = 2$. The AWE approximate delay (without the unstable poles, $v = 8$) is

$$\hat{T}_D = 40.07126238s \text{ with } error = 1.3230 \tag{4.20}$$

Table 4.1 Approximate poles and corresponding residues using the AWE method with $q = 10$ (Chebyshev low-pass filter, $N = 42$ and $A_{\max} = 0.1$ dB)

Approximate Poles	Approximate Residues
$-0.06539137 \pm j0.23262857$	$-0.02906163 \pm j0.04756440$
$-0.06139439 \pm j0.03746223$	$0.00561033 \pm j0.09507528$
$-0.06105120 \pm j0.11217647$	$-0.01669708 \pm j0.09319121$
$-0.06088147 \pm j0.18706118$	$0.03149358 \pm j0.09622437$
The unstable poles in the RHP: <u>$0.00567843 \pm j0.27547821$</u>	$0.00865480 \pm j0.00286852$

Then interpolation methods are used to compute the restored poles which are used to remedy the effect of the unstable poles in the RHP. By knowing that all poles are conjugated, only the poles with nonnegative imaginary parts are considered. Based on the different interpolation methods such as Lagrange, Cubic-spline and Hermite methods, several curves are created to join the useable approximate poles with nonnegative

Chapter 4: Improvement of Approximate Delay Accuracy by the Addition of Extra Poles

imaginary parts. In this way, it can be found that there is a relationship between the real parts of the approximate poles and their imaginary parts. Thus, the restored poles whose imaginary parts are the same as those of the poles in the RHP can be found on these interpolated functions. The corresponding conjugated restored poles can then be obtained.

Through Lagrange, Cubic-Spline and Hermite interpolations, the restored poles (Fausett 1999) can be computed as (i.e. $g = 2$): $-0.07543178 \pm j0.27547821$, $-0.07552433 \pm j0.27547821$, and $-0.06807234 \pm j0.27547820$, respectively. After obtaining new approximate pole collection including the obtained restored poles, the new approximate residues can be obtained.

For example, Table 4.2 gives the new approximate poles ($\nu = 10$) and residues after restoring the poles from Hermite interpolation. In the table the poles in bold is the restored poles. Then based on the new approximate poles and residues after pole restoration, new approximate delays are computed and are shown in Table 4.3.

Table 4.2 Approximate poles and corresponding residues after Hermite-interpolation pole restoration (Chebyshev low-pass filter, $N = 42$ and $A_{\max} = 0.1$ dB)

Approximate Poles	Approximate Residues
$-0.06539137 \pm j0.23262857$	$-0.05799623 \pm j0.07950916$
$-0.06139439 \pm j0.03746223$	$0.00561228 \pm j0.09507509$
$-0.06105120 \pm j0.11217647$	$-0.01679038 \pm j0.09302779$
$-0.06088147 \pm j0.18706118$	$0.03911866 \pm j0.10204541$
The restored poles: $-0.06807234 \pm j0.27547821$	$0.03005724 \pm j0.02405232$

Table 4.3 Approximate delays after pole restoration (Chebyshev low-pass filter, $N = 42$ and $A_{\max} = 0.1$ dB) (Cf.: Exact delay, $T_D = 41.39427213$ s and AWE approximate delay, $\hat{T}_D = 40.07126238$ s with $error = 1.3230$)

Interpolation Method	Restored Poles	Approximate Delay, $\hat{T}_D$ (s)	$error$ (s)	Error Rate φ
Lagrange	$-0.07543178 \pm j0.27547821$	40.44164047	0.9526	2.3013%
Cubic-Spline	$-0.07552433 \pm j0.27547821$	40.43797866	0.9563	2.3102%
Hermite	$-0.06807234 \pm j0.27547821$	40.51211408	0.8822	2.1312%

Chapter 4: Improvement of Approximate Delay Accuracy by the Addition of Extra Poles

From the restored poles in Table 4.3, it is observed that the real-part values do not differ very much. Also, it is found that the choice of interpolation method is not critical. Comparing the errors of new approximate delays in Table 4.3 with that of the AWE delay shown in eqn. (4.20), it is obvious that the new approximate delays have improved over to the AWE delay before pole restoration.

(B) RC – Tree Interconnect Network

Consider the RC-tree interconnect network shown in Figure 4.1, which has m branches and n RC cells. The cell number of the k^{th} branch is χ_k (assume that $\chi_1 \leq \chi_2 \leq \dots \leq \chi_k \leq \dots \leq \chi_m$) and $\alpha = C_t / C$, The number of RC cells is given by $n = 1 + \sum_{k=1}^m \chi_k$, which is also the overall order of the network i.e. $N = n$. In this example, we choose the branch number $m = 2$ (i.e. the base ladder structure plus one branch above), $\chi_1 = 14$, $\chi_2 = 15$, $\alpha = 0$ (i.e. $C_t = 0$), $R = 1\Omega$, $C = 10$ pF and $n = 30$.

Figure 4.1 An n -cell (n^{th} -order) RC-tree network with capacitive load

Chapter 4: Improvement of Approximate Delay Accuracy by the Addition of Extra Poles

It is found that the approximate poles obtained by the AWE technique contain some poles in the RHP when the approximate pole number, q , is beyond 7. The results listed in Table 4.4 show that the number of RHP poles increases with increasing approximate pole number, q .

Table 4.4 Approximate poles of RC-tree network using the AWE method ($n=30$, $m=2$, $\alpha=0$, $\chi_1=14$, and $\chi_2=15$.)

Network Order, N	Approximate Pole Number, q	AWE Approximate Poles (Poles underlined are in RHP)	Useable Poles Number, v
30 ($N=n$)	8	-0.40754807 -0.14835856 ± $j0.10019078$ -0.03020450 -0.01051139 ± $j0.00075724$ <u>0.02774624 ± $j0.03681997$</u>	6
	11	-0.24994710 ± $j0.43298156$ -0.04150764 ± $j0.04456043$ -0.00909017 -0.00523029 <u>0.00067611 ± $j0.00935458$</u> <u>0.00794838</u> <u>0.04396467 ± $j0.04086799$</u>	6
	16	-0.98164378 -0.01538558 ± $j0.42008352$ -0.01227574 ± $j0.00717104$ -0.00561904 ± $j0.00122223$ -0.00058705 ± $j0.00721833$ <u>0.00446617 ± $j0.01110046$</u> <u>0.00541424 ± $j0.00277922$</u> <u>0.00583253 ± $j0.03289954$</u> <u>0.01795272</u>	9

Because of the effect of deleting the poles in the RHP, the AWE approximate delay cannot reach the expected accuracy. Setting q to 11, 12 and 16, the AWE approximate delays are computed and compared with the exact delay obtained by the State-Space method. The results are shown in Table 4.5. In Table 4.5, v is the number of useable approximate poles.

Chapter 4: Improvement of Approximate Delay Accuracy by the Addition of Extra Poles

Table 4.5 Comparison of the exact delay and AWE approximate delays of the RC-tree network ($n=30$, $m=2$, $\alpha=0$, $\chi_1=14$, and $\chi_2=15$,)

State-Space Method	The AWE Technique				
Exact Delay T_D (s)	q	v	Approximate Delay, $\hat{T}_D$ (s)	<i>error</i> (s)	Error Rate ϕ
113.79507487	11	6	130.89930977	17.1042	13.0667%
	12	8	91.00772599	22.7873	20.0248%
	16	9	68.07530222	45.7198	40.1773%

From Table 4.4, it is observed that, when the network is approximated by an 11th-order system using the AWE, $r=5$ unstable poles and $v=q-r=6$ stable poles are produced. There is a real pole ($pole=0.00794838$) lying in the RHP. In this case, the restored pole can be regarded as the middle point between two useable real poles and is computed as

$$pole = \frac{-0.00909017 - 0.00523029}{2} = -0.00716023$$

Considering only the poles with nonnegative imaginary parts, the useable approximate poles obtained from the AWE method with $q=11$ and the corresponding interpolation curves are shown in Figure 4.2. Based on the imaginary part of original unstable poles shown in Table 4.4 and different interpolation methods, several sets of restored poles are obtained on the curves and listed in Table 4.6.

Table 4.6 Restored poles from different interpolation methods (RC-tree network; AWE method with $q=11$)

Interpolation Method	Restored Poles (Poles underlined are very close to original useable poles)
Lagrange	<u>-0.03996087</u> $\pm j0.04086799$ -0.02244605 $\pm j0.00935458$ -0.00716023
Cubic-Spline	<u>-0.03996087</u> $\pm j0.04086799$ -0.02244605 $\pm j0.00935458$ -0.00716023
Hermite	<u>-0.03919444</u> $\pm j0.04086799$ -0.02096764 $\pm j0.00935458$ -0.00716023

Figure 4.2 Three interpolation curves joining useable approximate poles from the AWE method with $q=11$ (RC-tree network)

Observing Figure 4.2 and Table 4.6, it can be found that, in this example, Lagrange and Cubic-spline methods cause the same interpolation curve and the same new approximate poles. Also, for avoiding the repeated effect, the restored poles very close to the original useable approximate poles (underlined in Table 4.6), are discarded. After adding a pair of conjugated restored poles or a real restored pole into the original useable AWE poles, the new approximate delays are obtained and are shown in Table 4.7. Then, adding all restored poles into the useable AWE poles, the corresponding new approximate delays are computed and listed in Table 4.8. In Tables 4.7 and 4.8, ν is the number of useable approximate poles.

Comparing the data in Table 4.5 and Table 4.7, it is obvious that after adding a pair of conjugated restored poles or a real restored pole to remedy the loss caused by the poles in the RHP, the new approximate delays are much more accurate than the original approximate delay obtained directly from the AWE technique. From Table 4.8, it is found

Chapter 4: Improvement of Approximate Delay Accuracy by the Addition of Extra Poles

that the accuracy of new approximate delay is visibly enhanced with adding more restored poles to replace the unstable poles.

Table 4.7 Approximate delays by adding a part of restored poles into the usable AWE poles (*RC*-tree network; AWE method with $q=11$) (Cf.: Exact delay, $T_D = 113.79507487\text{s}$; AWE approximate delay, $\hat{T}_D = 130.89930977\text{s}$ with $\text{error} = 17.1042$)

Interpolation Method	Restored Pole	ν	Approximate Delay, $\hat{T}_D$ (s)	error (s)	Error Rate φ
Lagrange & Cubic-spline	-0.02244605 $\pm j0.00935458$	8	120.31882596	6.5238	5.7329%
	-0.00716023	7	118.85414795	5.0591	4.4458%
Hermite	-0.02096764 $\pm j0.00935458$	8	121.96882928	8.1738	7.1829%
	-0.00716023	7	118.85414795	5.0591	4.4458%

Table 4.8 Approximate delays by adding all restored poles into the usable AWE poles (*RC*-tree network; AWE method with $q=11$) (Cf.: Exact delay, $T_D = 113.79507487\text{s}$; AWE approximate delay, $\hat{T}_D = 130.89930977\text{s}$ with $\text{error} = 17.1042$)

Interpolation Method	Restored Pole	ν	Approximate Delay, $\hat{T}_D$ (s)	error (s)	Error Rate φ
Lagrange & Cubic-spline	-0.02244605 $\pm j0.00935458$	9	114.74707321	0.9520	0.8366%
	-0.00716023				
Hermite	-0.02096764 $\pm j0.009354578$	9	115.48509299	1.6900	1.4851%
	-0.00716023				

Hence, interpolation methods are effective to increase the useable approximate pole number, remedy the loss caused by discarded poles and bring better accuracy. Observing Tables 4.7 and 4.8, it is also found that the choice of interpolation method is not critical.

(C) *RLC*–Tree Interconnect Network

Consider the capacitively loaded *RLC*-tree interconnect network shown in Figure 4.3 where the general parameters are defined similarly as in Figure 4.1 (*RC*-tree network). Here, the number of *RLC* cells is given by $n = 1 + \sum_{k=1}^m \chi_k$ and the overall network order N is equal to $2n$. The parameters are set: $m = 2$, $\chi_1 = 8$, $\chi_2 = 11$, $\alpha = 13$, $n = 20$, $R = 1\Omega$, $L = 1 \text{ nH}$, $C = 10\text{pF}$.

Chapter 4: Improvement of Approximate Delay Accuracy by the Addition of Extra Poles

Figure 4.3 An n -cell ($2n^{\text{th}}$ -order) RLC-tree network with capacitive load

When $q > 3$, the AWE technique produces some unstable poles in the RHP. In Table 4.9, the AWE poles with different q are listed and the unstable poles are underlined. Setting q to 24 and 30, the AWE approximate delays are compared with the exact delay from the state-space method. The results are shown in Table 4.10, where the “error” and the corresponding error rate ϕ have the same definitions as eqn. (4.17) and (4.18).

The interpolation methods are then used to compute the new poles for restoration. Considering only the poles with nonnegative imaginary parts, the useable approximate poles obtained from the AWE method with $q=24$ and the corresponding interpolation curves are shown in Figure 4.4. Then the corresponding restored poles, whose imaginary parts are the same as those in the RHP, are obtained from these curves and presented in Table 4.11. In the table, the underlined poles are very close to some original useable poles and can be regarded as repeated poles. Therefore these poles will not be used as restored

Chapter 4: Improvement of Approximate Delay Accuracy by the Addition of Extra Poles

poles. After using the restored poles to remedy the effect of unstable poles in the RHP, the new approximate delays are obtained and listed in Table 4.12.

Table 4.9 Approximate poles of *RLC*-tree network using the AWE method ($n=20$, $m=2$, $\alpha=13$, $\chi_1=8$ and $\chi_2=11$)

Network Order, N	Approximate Pole Number, q	AWE Approximate Poles (Poles underlined are in RHP)	Useable Poles Number, v
40 ($N=2n$)	4	-119.43046236 -8.50604224 -0.10207115 <u>0.03857575</u>	3
	14	-570.61634758 -5.38524115 -0.05502004 -0.04394888 $\pm j0.04819291$ -0.02289840 $\pm j0.03069630$ -0.00101286 $\pm j0.09725163$ <u>0.01624671 $\pm j0.05293382$</u> <u>0.04998949 $\pm j0.03840151$</u> <u>0.05985666</u>	9
	24	-297.51928182 -1.16255803 -0.13839488 -0.07369881 -0.06388142 $\pm j0.01969182$ -0.04990015 $\pm j0.03913232$ -0.02970097 $\pm j0.05373569$ -0.01477579 -0.00025985 $\pm j0.06222518$ -0.00351279 $\pm j0.07579244$ <u>0.13162360</u> <u>0.03022922 $\pm j0.05912621$</u> <u>0.04196386 $\pm j0.05035837$</u> <u>0.06279343 $\pm j0.03006901$</u> <u>0.06747820 $\pm j0.00911253$</u>	15

Table 4.10 Comparison of the exact delay and AWE approximate delays of the *RLC*-tree network ($n=20$, $m=2$, $\alpha=13$, $\chi_1=8$ and $\chi_2=11$)

State-Space Method	AWE Technique				
Exact Delay T_D (s)	q	v	Approximate Delay, $\hat{T}_D$ (s)	<i>error</i> (s)	Error Rate φ
10.90243906	24	15	11.03908194	0.1366	1.2533%
	30	17	10.72790863	0.1745	1.6008%

Figure 4.4 Three interpolation curves joining useable approximate poles obtained from the AWE method with $q=24$ (Except three real useable poles smaller than -0.07369881 ; RLC -tree network)

Table 4.11 Restored poles from different interpolation methods (RLC -tree network; AWE method with $q=24$)

Interpolation Method	Restored Poles (Poles underlined are very close to original useable poles)
Lagrange	$-0.07268993 \pm j0.00911253$
	$-0.05215549 \pm j0.03006901$
	$-0.04734412 \pm j0.05035837$
	<u>$-0.00628714 \pm j0.05912621$</u>
Cubic-spline	$-0.06970597 \pm j0.00911253$
	$-0.05544926 \pm j0.03006901$
	$-0.04528895 \pm j0.05035837$
	<u>$-0.00716380 \pm j0.05912621$</u>
Hermite	$-0.06829437 \pm j0.00911253$
	$-0.05727271 \pm j0.03006901$
	$-0.03794057 \pm j0.05035837$
	<u>$-0.00692639 \pm j0.05912621$</u>

Chapter 4: Improvement of Approximate Delay Accuracy by the Addition of Extra Poles

Table 4.12 Approximate delays by adding a pair of restored poles into the usable AWE poles (RLC-tree network; AWE method with $q=24$) (Cf.: Exact delay, $T_D = 10.90243906\text{s}$; AWE approximate delay, $\hat{T}_D = 11.03908194\text{s}$ with $error = 0.1366$)

Interpolation Method	Restored Pole	ν	Approximate Delay, $\hat{T}_D$ (s)	$error$ (s)	Error Rate φ
Lagrange	-0.07268993 $\pm j0.00911253$	17	10.89262745	0.0098	0.0900%
	-0.05215549 $\pm j0.03006901$	17	10.90897368	0.0065	0.0599%
	-0.04734412 $\pm j0.05035837$	17	10.94130029	0.0389	0.3564%
Cubic-spline	-0.06970597 $\pm j0.00911253$	17	10.89239142	0.0100	0.0922%
	-0.05544926 $\pm j0.03006901$	17	10.90653485	0.0041	0.0376%
	-0.04528895 $\pm j0.05035837$	17	10.94494096	0.0425	0.3898%
Hermite	-0.06829437 $\pm j0.00911253$	17	10.89232350	0.0101	0.0928%
	-0.05727271 $\pm j0.03006901$	17	10.90540351	0.0030	0.0272%
	-0.03794057 $\pm j0.05035837$	17	10.96053479	0.0581	0.5329%

Comparing Table 4.10 and Table 4.12, it is found that after adding a pair of restored poles, the accuracy of approximate delay is greatly improved. Also, it is observed that the new approximate delay is more accurate than the original AWE approximate delay ($q = 24$, $\nu = 15$) and the AWE delay from the same number of useable approximate poles ($q = 30$, $\nu = 17$).

Hence, the conclusion is strengthened that interpolation methods are effective to increase the accuracy of approximate delay by adding new restored poles to replace the poles in the RHP.

4.4.2 Delay Evaluation Using Extended Two-Pole Method

Considering the same interconnect network including transmission lines shown in Figure 3.8, it can be transformed into an 8th-order equivalent circuit only including linear elements as shown in Figure 3.9. The exact delay of the equivalent circuit obtained from the state-space method and the AWE approximate delay with $q = 2$ are listed in Table 4.13. In Table 4.13, the “error” is defined as in eqn. (4.17).

Table 4.13 The exact delay and AWE approximate delays of the example interconnect network

State-Space Method	AWE Technique		
Exact Delay, T_D (s)	q	Approximate Delay, $\hat{T}_D$ (s)	error (s)
0.67210336178693	2	0.47019670104191	0.2019

As shown in Section 3.7.2, the moments of circuit, m_j , can be obtained. After that, α_{m_j} and β_{m_j} are given by eqn. (4.11). All data are shown in Table 3.13 in Section 3.7.2. After illustrating the values of β_{m_j} in the Cartesian coordinates with j being the horizontal axis and β_{m_j} being the vertical axis, the fluctuation of β_{m_j} resembles a wave shape. Thus, the β wave is obtained.

Three crests of the β wave can be found as β_{m_3} , β_{m_7} and β_{m_11} , and every crest of the β wave corresponds to a pair of approximate poles. Computed approximate delays are listed in Table 4.14. Besides, the step responses are compared in Figure 4.5.

Table 4.14 Approximate delay from different crests (“error” is defined as in eqn. (4.17)) (Cf: Exact delay, $T_D = 0.67210336178693$ s)

Crest	Two Approximate Poles	Approximate Delay $\hat{T}_D$ (s)	error (s)
β_{m_3}	$-2.29845694019409 \pm j2.56618157548245$	0.40765454855683	0.2644
β_{m_7}	$-1.84770187869371 \pm j2.22571188991250$	0.47858708495100	0.1935
β_{m_11}	$-1.70213767287625 \pm j1.32085223880269$	0.69466300887063	0.0226

Figure 4.5 Comparison of step responses using the state-space, AWE and two-pole methods (Two approximate poles are used)

Here, the two-pole method is extended to use more than one pair of approximate poles. If two crest values of the β wave are applied, two pairs of approximate poles can be computed. Then the approximate delay can be computed. Applying any two crests, the results are presented in Table 4.15. Next, three pairs of approximate poles from three crests are applied to compute the corresponding delay which is also shown in Table 4.15 and the “error” is defined as in eqn. (4.17).

Comparing the data in Tables 4.14 and 4.15, it is observed that the accuracy of the approximate delay increases with the number of approximate poles. However, this may not be true in general. Since every crest stands for a pair of approximate poles, if more crests are applied, their contributions accumulate and result in a more accurate approximate delay.

When q is set to 6, it is shown in Table 4.16 that the AWE method has two unstable poles in the right-half s -plane (RHP) and should be discarded (Chiprout and Nakhla 1994).

Chapter 4: Improvement of Approximate Delay Accuracy by the Addition of Extra Poles

Thus, only four useable approximate poles are applied to compute the delay. If $q = 8$, the useable pole number is still 4. Until q is set to 14, the number of useable poles is 6. It is noticed that with q increasing, more and more unstable poles are produced. This is because of the numerical instability of the AWE method. All results are listed in Table 4.16.

Table 4.15 Approximate delays using more than one crest of the β wave (Cf: Exact delay, $T_D = 0.67210336178693$ s,)

Applied Crest Number	Applied Crests	Approximate Delay, $\hat{T}_D$ (s)	error (s)
2	β_{m_3} and β_{m_7}	0.65925699361983	0.0128
2	β_{m_3} and β_{m_11}	0.62756525395855	0.0445
2	β_{m_7} and β_{m_11}	0.63010911470656	0.0420
3	β_{m_3} , β_{m_7} and β_{m_11}	0.66671916277589	0.0054

Figure 4.6 compares the step responses of the state-space (exact), AWE and extended two-pole methods. The extended two-pole method uses three crests of the β wave and the AWE method has $q = 6$. Here, PSpice is used to verify the exact step response from the state-space method.

From Figure 4.6, it is found that the extended two-pole method has a better accuracy than the AWE method having the same pole number. Similar observation is also found in Tables 4.15 and 4.16. From Table 4.15, the approximate delay using the extended two-pole method with three crests, β_{m_3} , β_{m_7} and β_{m_11} (i.e. β_{crest_1} , β_{crest_2} and β_{crest_3}) i.e. six approximate poles, is

$$\hat{T}_D = 0.66671916277589 \text{ s and } error = 0.0054 \text{ s}$$

From Table 4.16, the AWE method ($q = 12$) with six useful approximate poles has

$$\hat{T}_D = 0.64397054631797 \text{ s and } error = 0.0281 \text{ s}$$

The extended two-pole method has a better performance in this case.

Chapter 4: Improvement of Approximate Delay Accuracy by the Addition of Extra Poles

Table 4.16 AWE approximate poles and delay with different pole number

q	$q_{uesable}$	AWE Approximate Poles (The poles underlined are in RHP)	Approximate Delay, $\hat{T}_D$ (s)	$error$ (s)
4	4	-2.10494668336906 ± j 4.64896776573669 -1.98745476152813 ± j 2.38374046431969	0.67210336178789	9.6000e-13
6	4	-1.60681394147411 ± j 5.29629789487574 -2.12470643780574 ± j 1.92787444578658 <u>2.30294895070841 ± j1.99481073399603</u>	0.63261000728687	0.0395
8	4	-1.42392532581174 ± j 4.01843262601546 -2.17789058846233 ± j 0.16029584852526 <u>1.74206080255790 ± j3.90611696831300</u> <u>3.33183175199641</u> <u>0.38767847143592</u>	0.64466710974784	0.0274
12	6	-0.36005444239285 ± j 2.91841719152984 -2.24967658972098 ± j 1.84592656733209 -2.49024746686884 ± j 5.23343307486259 <u>0.12855789987796</u> <u>0.76892710123602</u> <u>1.98758614421486 ± j2.11785549056019</u> <u>5.61089095610660</u> <u>29.15640875231505</u>	0.64397054631797	0.0281
14	6	-1.43528877826530 ± j 2.02327733695425 -1.94788775635955 ± j 3.15212489912167 -2.83547621447236 -3.59291965385774 <u>0.92304836621399 ± j2.47664342661622</u> <u>1.38082104984250 ± j4.30823136228646</u> <u>2.20044938850966 ± j0.93177107626679</u> <u>3.84648032175799 ± j2.47673603006577</u>	0.65201754893423	0.0201
18	7	-1.26188906872904 ± j 2.91458754539604 -1.81766775808710 ± j 2.36402119757567 -2.69576761049460 ± j 1.38788628618973 -2.77757173134105 <u>-10.20710641200123 ± j5.85697764385453</u> <u>0.59628861572730 ± j3.24217974953806</u> <u>1.66938618256028 ± j2.57078098917324</u> <u>2.24628185123977 ± j2.32772592926106</u> <u>2.71074685410199 ± j0.87206095231328</u> <u>3.70443383011373</u>	0.65825140777065	0.0139

Figure 4.6 Comparison of step responses using the state-space, AWE, extended two-pole methods and PSpice. (Six approximate poles are used)

The AWE and extended two-pole methods were compared for their execution time and the results are shown in Table 4.17. It can be seen that the extended two-pole method is more efficient.

Table 4.17 Efficiency Comparison of AWE and extended two-pole methods

AWE Method	Approximate Pole Number		Computation Time (s)
	2		4.665
	4		6.929
	6		13.810
Extended Two-Pole Method	Crest	Approximate Pole Number	Computation Time (s)
	1	2	2.352
	2	4	5.280
	3	6	8.705

4.5 Summary

The AWE technique is an accurate approximation method based on moment matching technology which is limited to its inherent numerical instability. In high-order networks, the AWE method will generate some unstable approximate poles in the right-half s -plane (RHP) and these unstable poles must be discarded. In this chapter, a heuristic method is presented to apply interpolation method to produce new approximate poles in the left-half s -plane (LHP). These LHP poles are then used to replace those removed poles. Comparing the AWE delay with the AWE-based delay by pole interpolation, it is found that interpolation methods can be effectively used to restore poles to increase the useable pole number and enhance the delay accuracy.

On another issue, although the two-pole method is more efficient than the AWE method, its accuracy is much lower. An extended two-pole method is proposed to use more crests in the β wave to produce more pairs of dominant approximate poles. These dominant poles are then used a new to create a new approximation model in order to compute the new approximate delay. Comparing the extended two-pole method with the AWE method, the extended two-pole method has a better accuracy and shorter computation time than the AWE method having the same pole number.

Chapter 5: Delay Time Bounds for RC- and RLC-Cell Networks

5.1 Introduction

With the ever-increasing demand for faster logic devices and higher capacity memories, the minimum feature size has been scaled to submicron level (Hanchate and Ranganathan 2006). As a result, the interconnect performance has become a limiting factor that affects circuit performances. In VLSI designs, the delay introduced by the interconnections may be comparable to or longer than active-device delay. Thus, upper and lower bounds for delay are computed and presented in this chapter. The results can be used: (1) to bound the delay, given the signal threshold, or (2) to bound the signal voltage, given a delay time, or (3) to certify that a circuit is “fast enough,” given both the maximum delay and the voltage threshold.

Marinov and Shivakumar (1996) defined the global switching time (GST) and obtained the GST bounds of the RC-ladder network shown in Figure 5.1, a basic model in integrated circuits. More details about the definition of the GST are given in Section. 5.3.1. The network is composed of n RC-cells, capacitively loaded by C_i , and excited at $t = 0$ by a constant step voltage source v_s . Choosing the corresponding GST parameter λ , the network delay time can be regarded as a special case of the GST, and thus the network delay bound can be computed from the GST.

Since the RC-tree structure is widely used in VLSI circuit design, similar delay bound computation method is extended to the RC-tree network shown in Figure 5.2 (same as Figure 4.1 and repeated for convenience). The branches of the network in Figure 5.2 are arranged with the branch cell number increasing. In other words, it has the following relationship:

$$\chi_1 \leq \chi_2 \leq \cdots \leq \chi_k \leq \cdots \leq \chi_m \quad (5.1)$$

where χ_k is the cell number of the k^{th} branch, m is the network branch number, and

n is the total cell number of the network given by

$$n = 1 + \sum_{k=1}^m \chi_k \quad (5.2)$$

Figure 5.1 An n -cell RC-ladder network with capacitive load

Figure 5.2 An n -cell RC-tree network with capacitive load (same as Figure 4.1 and repeated for convenience)

Chapter 5: Delay Time Bounds for RC - and RLC -Cell Networks

Furthermore, the method can also be extended for computing the delay bounds of RLC -ladder and RLC -tree networks which have the structures shown in Figure 5.3 and Figure 5.4 respectively. In these networks, the parameters C_t , χ_k , m and n have the same definitions as in RC -tree networks in Figure 5.2. For RLC -tree networks, eqns. (5.1) and (5.2) are also applicable.

Figure 5.3 An n -cell RLC -ladder network with capacitive load (u is a scaling factor)

Figure 5.4 An n -cell RLC -tree network with capacitive load

Chapter 5: Delay Time Bounds for RC- and RLC-Cell Networks

Comparing the *RLC*-tree networks in Figure 4.3 (Chapter 4) and Figure 5.3, it is observed that the network in Figure 5.3 is a general case but that in Figure 4.3 is a special case when $u = 10$. In general, a *RC*- or *RLC*-ladder network can be regarded as a special case of *RC*- or *RLC*-tree network with a unique branch ($m = 1$).

5.2 System Description

For a chosen GST parameter λ of a network, the GST can be transformed to the system delay time as explained in Section 5.4. The following sections present the GST evaluations of four types of networks shown in Figures 5.1-5.4, respectively. Considering the structures of different networks, the relationship between the network order N and the network cell number n is shown as

$$\begin{aligned} N &= n & (\text{RC-ladder/tree network}) \\ N &= 2n & (\text{RLC-ladder/tree network}) \end{aligned} \quad (5.3)$$

and the ratio of C_i and C is defined as

$$\alpha = C_i / C \quad (5.4)$$

where C_i is shown in Figures 5.1-5.4.

5.2.1 System Characterization of RC-Ladder Networks

The characteristics of a two-port interconnect with single input and single output can be described by the following state equations:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{b}u(t) \\ y(t) &= \mathbf{c}^T \mathbf{x}(t) + du(t) \end{aligned} \quad (5.5)$$

where the definitions of parameters are given in Section 3.2.1. Given the n -cell *RC*-ladder network shown in Figure 5.1 which is excited at $t = 0$ by a constant voltage source v_s , the system state vector $\mathbf{x}(t)$ is the n -dimensional voltage vector of the capacitors,

$$\begin{cases} \mathbf{x}(t) = [v_1(t), v_2(t), \dots, v_n(t)]^T \\ \mathbf{x}(0) = \mathbf{x}^0 \quad \text{and} \quad \mathbf{x}(t \rightarrow \infty) = \mathbf{x}^\infty \end{cases} \quad (5.6)$$

where $\mathbf{x}^0$ is the initial network state vector, and $\mathbf{x}^\infty$ is the steady network state vector.

The first equation of eqn. (5.5) can be transformed to

$$\dot{\mathbf{x}}(t) = -\mathbf{A}_t \mathbf{x}(t) + [v_s/RC, 0, 0, \dots, 0]^T_{1 \times n} \quad (5.7)$$

where $\mathbf{A}_t$ is an $n \times n$ square matrix given by

$$\mathbf{A}_t = -\mathbf{A} = \frac{1}{RC} \begin{bmatrix} & & & & & 0 \\ & & & & & 0 \\ & & & & & \vdots \\ & & & \mathbf{J} & & \vdots \\ & & & & & 0 \\ & & & & & -1 \\ 0 & 0 & \dots & \dots & 0 & \frac{-1}{1+\alpha} & \frac{1}{1+\alpha} \end{bmatrix}_{n \times n} \quad (5.8)$$

with

$$\mathbf{J} = \begin{bmatrix} 2 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & -1 & 2 & -1 & & \\ & & -1 & \ddots & & \\ & & & \ddots & -1 & \\ & & & -1 & 2 & -1 \\ & & & & -1 & 2 \end{bmatrix}_{(n-1) \times (n-1)} \quad (5.9)$$

5.2.2 System Characterization of RC-Tree Networks

For the n -cell RC-tree network with m branches shown in Figure 5.2 and denoting the network state vector as in eqn. (5.6), eqn. (5.7) is obtained. For the RC-tree network, $\mathbf{A}_t$ is also an $n \times n$ square matrix having the same structure as eqn. (5.8). However, $\mathbf{J}$ is given as a $(n-1) \times (n-1)$ square matrix:

$$\mathbf{J} = \begin{matrix} 1 \\ 2 \\ \vdots \\ \vdots \\ \chi_1+1 \\ \vdots \\ 2+\sum_{i=1}^{k-1}\chi_i \\ \vdots \\ \vdots \\ 1+\sum_{i=1}^k\chi_i \\ \vdots \\ 2+\sum_{i=1}^{m-1}\chi_i \\ \vdots \\ n-2 \\ n-1 \end{matrix} \begin{bmatrix} m+1 & -1 & & -1 & & -1 \\ -1 & 2 & -1 & & & \\ & -1 & \ddots & \ddots & & \\ & & \ddots & 2 & -1 & \\ & & & -1 & 1 & \\ & & & & \ddots & 0 \\ -1 & & & 0 & 2 & -1 \\ & & & -1 & \ddots & \ddots \\ & & & & \ddots & 2 & -1 \\ & & & & & -1 & 1 \\ & & & & & & \ddots & 0 \\ & -1 & & & & 0 & 2 & -1 \\ & & & & & -1 & \ddots & \ddots \\ & & & & & & \ddots & 2 & -1 \\ & & & & & & & -1 & 2 \end{bmatrix} \quad (5.10)$$

5.2.3 System Characterization of RLC-Ladder Networks

Consider the n -cell RLC-ladder network with a constant voltage excitation in Figure 5.3 and denote the network state vector by

$$\begin{cases} \mathbf{x}(t) = [x_1(t), x_2(t), \dots, x_{2n-1}(t), x_{2n}(t)]^T \\ \quad = [i_1(t), v_1(t), i_2(t), v_2(t), \dots, i_n(t), v_n(t)]^T \\ \mathbf{x}(0) = \mathbf{x}^0 \quad \text{and} \quad \mathbf{x}(t \rightarrow \infty) = \mathbf{x}^\infty \end{cases} \quad (5.11)$$

which is composed of the voltages of the capacitors and the currents of the inductors; the dynamic evolution of the network can be derived as

$$\dot{\mathbf{x}}(t) = -\mathbf{A}_t \mathbf{x}(t) + [v_s/L, 0, 0, \dots, 0]_{1 \times 2n}^T \quad (5.12)$$

where

$$\mathbf{A}_t = -\mathbf{A} = X \begin{bmatrix} & & & & & 0 \\ & & & & & 0 \\ & & & & & \vdots \\ & & & & & \vdots \\ & & & & & \vdots \\ & & & & & 0 \\ & & & & & 1 \\ 0 & 0 & \dots & \dots & 0 & \frac{-1}{1+\alpha} & 0 \end{bmatrix}_{2n \times 2n} \quad (5.13)$$

with

$$\mathbf{J} = \begin{bmatrix} R & 1 & & & & \\ -1 & 0 & 1 & & & \\ & -1 & uR & 1 & & \\ & & -1 & 0 & 1 & \\ & & & -1 & u^2R & 1 \\ & & & & -1 & 0 & \ddots \\ & & & & & \ddots & \ddots & \ddots \\ & & & & & & \ddots & u^{n-2}R & 1 \\ & & & & & & & -1 & 0 & 1 \\ & & & & & & & & -1 & u^{n-1}R \end{bmatrix}_{(2n-1) \times (2n-1)} \quad (5.14)$$

Here, $\mathbf{A}_t$ is an $2n \times 2n$ square matrix whose components are $(\mathbf{A}_t)_{i,j}$, that is the i^{th} -row and j^{th} -column element of $\mathbf{A}_t$. The multiplicative operator X , that multiple every component of $\mathbf{A}_t$, is decided by the row position of the corresponding component and is shown as

$$X = \begin{cases} 1/L & \text{when } i \text{ is odd} \\ 1/C & \text{when } i \text{ is even} \end{cases} \quad (5.15)$$

5.2.4 System Characterization of RLC-Tree Networks

For the n -cell RLC-tree network with m branches in Figure 5.4 and with the network state vector as in eqn. (5.11), we can derive eqn. (5.12). For the RLC-tree network, $\mathbf{A}_t$ is also a $2n \times 2n$ square matrix whose components are $(\mathbf{A}_t)_{i,j}$ and can be derived as

$$\mathbf{A}_t = -\mathbf{A} = X \begin{bmatrix} R & 1 & 0 & \cdots & 0 \\ -1 & & & & \vdots \\ 0 & & \mathbf{J}_m & & 0 \\ \vdots & & & & 1 \\ 0 & \cdots & 0 & \frac{-1}{(1+\alpha)} & 0 \end{bmatrix}_{2n \times 2n} \quad (5.16)$$

where the multiplicative operator X is same as in eqn. (5.15) and $\mathbf{J}_m$ (m is the network branch number) is described by the following recursive relationship:

Chapter 5: Delay Time Bounds for RC- and RLC-Cell Networks

$$\begin{cases} \mathbf{J}_1 = \mathbf{P}_1 & k=1 \\ \mathbf{J}_k = \begin{bmatrix} \mathbf{J}_{k-1} & \mathbf{M}_k \\ \mathbf{N}_k & \mathbf{P}_k \end{bmatrix} & k \in [2, m] \end{cases} \quad (5.17)$$

where $\mathbf{J}_k$ is an $N_{\mathbf{J}_k} \times N_{\mathbf{J}_k}$ square matrix with

$$\begin{cases} N_{\mathbf{J}_k} = 2n - 2 - 2 \sum_{i=k+1}^m \chi_i & k \in [1, m-1] \\ N_{\mathbf{J}_m} = 2n - 2 & k = m \end{cases} \quad (5.18)$$

and $\mathbf{M}_k$ ($N_{\mathbf{J}_{k-1}} \times 2\chi_k$) has zero entries except the two elements:

$$(\mathbf{M}_k)_{1,2} = (\mathbf{M}_k)_{N_{\mathbf{J}_{k-1}},1} = 1 \quad (5.19)$$

$\mathbf{N}_k$ can be represented as

$$\mathbf{N}_k = -\mathbf{M}_k^T \quad (5.20)$$

and $\mathbf{P}_k$ is a $2\chi_k \times 2\chi_k$ square matrix given by

$$\mathbf{P}_k = \begin{bmatrix} 0 & 0 & & & & & \\ 0 & uR & 1 & & & & \\ & -1 & 0 & 1 & & & \\ & & -1 & u^2R & \ddots & & \\ & & & \ddots & \ddots & 1 & \\ & & & & -1 & 0 & 1 \\ & & & & & -1 & u^{\chi_k}R \end{bmatrix}_{2\chi_k \times 2\chi_k} \quad k \in [2, m] \quad (5.21)$$

and

$$\mathbf{P}_1 = \begin{bmatrix} 0 & 1 & & & & & \\ -1 & uR & 1 & & & & \\ & -1 & 0 & 1 & & & \\ & & -1 & u^2R & \ddots & & \\ & & & \ddots & \ddots & 1 & \\ & & & & -1 & 0 & 1 \\ & & & & & -1 & u^{\chi_1}R \end{bmatrix}_{2\chi_1 \times 2\chi_1} \quad (5.22)$$

For example, a 7-cell RLC-tree network with 3 branches has $\chi_1 = 1$, $\chi_2 = 2$ and $\chi_3 = 3$;

$\mathbf{A}_t$ is computed by eqns. (5.16)-(5.22) and is given by

$$\mathbf{A}_t = \begin{bmatrix} \frac{R}{L} & \frac{1}{L} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\frac{1}{C} & 0 & \frac{1}{C} & 0 & \frac{1}{C} & 0 & 0 & 0 & \frac{1}{C} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{-1}{L} & \frac{uR}{L} & \frac{1}{L} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{-1}{C} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{-1}{L} & 0 & 0 & \frac{uR}{L} & \frac{1}{L} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{-1}{C} & 0 & \frac{1}{C} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{-1}{L} & \frac{u^2 R}{L} & \frac{1}{L} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{-1}{C} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{-1}{L} & 0 & 0 & 0 & 0 & 0 & 0 & \frac{uR}{L} & \frac{1}{L} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{-1}{C} & 0 & \frac{1}{C} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{-1}{L} & \frac{u^2 R}{L} & \frac{1}{L} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{-1}{C} & 0 & \frac{1}{C} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{-1}{L} & \frac{u^3 R}{L} & \frac{1}{L} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{-1}{C(1+\alpha)} & 0 \end{bmatrix}$$

5.3 Derivation of Global Switching Time Bounds

5.3.1 Method to Compute Global Switching Time

Let the network status vector be represented by

$$\begin{cases} \mathbf{z}(t) = \mathbf{x}(t) - \mathbf{x}^\infty \\ \mathbf{z}^0 = \mathbf{x}^0 - \mathbf{x}^\infty \end{cases} \quad (5.23)$$

where $\mathbf{z}^0$ is the initial status vector of the network, and the network state vector $\mathbf{x}(t)$ of the RC-cell and RLC-cell networks (including two bound cases: $\mathbf{x}^0$ and $\mathbf{x}^\infty$) are presented in eqn. (5.6) and (5.11), respectively. The dynamic behaviors of the RC- and RLC-cell networks can be characterized by the following linear system:

$$\begin{cases} \dot{\mathbf{z}}(t) = -\mathbf{A}_t \mathbf{z}(t) \\ \mathbf{z}(0) = \mathbf{z}^0 \end{cases} \quad (5.24)$$

The following index function defined by Marinov and Shivakumar (1996) is used:

$$f(t) = \|\mathbf{z}(t)\| / \|\mathbf{z}^0\| \quad (5.25)$$

with

$$\|\mathbf{z}(t)\| = \sum_{i=1}^N |z_i(t)| \quad (5.26)$$

where $z_i(t)$ is the i^{th} component of $\mathbf{z}(t)$, and the network order N of the RC- and RLC-cell networks is given in eqn. (5.3).

Thus, the global switching time of the network (denoted by T_λ) is defined to be the instant when $f(t)$ equals the value $\lambda \in (0,1)$. Since $f(t)$ is monotonically decreasing, it can be shown that

$$T_\lambda = f^{-1}(\lambda) \quad (5.27)$$

5.3.2 Construction of Global Switching Time Bounds

For a low order network, the exact value of T_λ can be easily computed. Therefore, only cases with $N \geq 4$ are considered here. Because $\mathbf{A}_t$ is irreducible diagonally dominant, its inverse exists. Furthermore, it is assured by the Perron-Frobenius theorem (Golub 1996) that $(\mathbf{A}_t^T)^{-1}$ has a positive eigenvalue denoted by $1/q_{ev}$, i.e. q_{ev} is the eigenvalue of $\mathbf{A}_t^T$. q_{ev} corresponds to an eigenvector $\mathbf{r}$ with positive components $r_k > 0$. This means that

$$\mathbf{A}_t^T \mathbf{r} = q_{ev} \mathbf{r} \quad (5.28)$$

With this equation, eqn. (5.24) immediately gives

$$\frac{d}{dt} \mathbf{r}^T \mathbf{z} = -q_{ev} \mathbf{r}^T \mathbf{z} \quad (5.29)$$

Assuming that

$$y_{ev}(t) = \mathbf{r}^T \mathbf{z} \quad (5.30)$$

eqn. (5.29) can then be transferred to

$$\frac{d}{dt} y_{ev}(t) = -q_{ev} y_{ev}(t) \quad (5.31)$$

Chapter 5: Delay Time Bounds for RC- and RLC-Cell Networks

To calculate the integral of eqn. (5.31), it can be obtained

$$\int_{t_1}^t \frac{dy_{ev}(t)}{dt} dt = \int_{t_1}^t dy_{ev}(t) = \int_{t_1}^t -q_{ev} y_{ev}(t) dt \quad (5.32)$$

where t_1 is an arbitrary value of t . It is easily found that eqn. (5.32) exists only when

$$y_{ev}(t) = c_{ev} e^{-q_{ev} t} \quad (5.33)$$

Arbitrarily taking $t = 0$ and substituting it into eqn. (5.33), we have

$$c_{ev} = y_{ev}(0) \quad (5.34)$$

Thus, eqn. (5.33) can be expressed as

$$y_{ev}(t) = y_{ev}(0) e^{-q_{ev} t} \quad (5.35)$$

Substituting eqn. (5.30) to eqn. (5.35), there is

$$\mathbf{r}^T \mathbf{z} = e^{-q_{ev} t} \mathbf{r}^T \mathbf{z}^0 \quad (5.36)$$

which can also be described by

$$\sum_{i=1}^N r_i z_i(t) = e^{-q_{ev} t} \sum_{i=1}^N r_i z_i^0 \quad (5.37)$$

Thus, t can be derived from eqns. (5.36) and (5.37) as

$$t = -\frac{1}{q_{ev}} \ln \frac{\mathbf{r}^T \mathbf{z}}{\mathbf{r}^T \mathbf{z}^0} = -\frac{1}{q_{ev}} \ln \frac{\sum_{i=1}^N r_i z_i(t)}{\sum_{i=1}^N r_i z_i^0} \quad (5.38)$$

Suppose that four positive numbers $\underline{q}_{ev}$, $\bar{q}_{ev}$, $\underline{r}$, $\bar{r}$ are the bounds of q_{ev} and r_k such that

$$\underline{q}_{ev} \leq q_{ev} \leq \bar{q}_{ev} \quad \text{and} \quad \underline{r} \leq r_k \leq \bar{r} \quad \text{for } k = 1, 2, \dots, N. \quad (5.39)$$

Then, the lower bound $\underline{T}_\lambda$ and upper bound $\bar{T}_\lambda$ of the GST can be computed from eqns. (5.27) and (5.38) as

$$\underline{T}_\lambda = \max \left\{ 0; -\frac{1}{\bar{q}_{ev}} \ln \frac{\lambda \bar{r}}{\underline{r}} \right\} \quad \text{and} \quad \bar{T}_\lambda = -\frac{1}{\underline{q}_{ev}} \ln \frac{\lambda \underline{r}}{\bar{r}} \quad (5.40)$$

Denoting A_{t-ij}^{-1} the elements of $\mathbf{A}_t^{-1}$ and normalizing the eigenvector $\mathbf{r}$ such that

$\sum_{k=1}^N r_k = 1$, eqn. (5.28) leads to

$$\min_i \sum_{j=1}^N |A_{t-ij}^{-1}| \leq \frac{1}{q_{ev}} \leq \max_i \sum_{j=1}^N |A_{t-ij}^{-1}| \quad (5.41)$$

$$\min_j \sum_{i=1}^N |A_{t-ij}^{-1}| \leq \frac{1}{q_{ev}} \leq \max_j \sum_{i=1}^N |A_{t-ij}^{-1}| \quad (5.42)$$

$$q_{ev} \min_{i,j} |A_{t-ij}^{-1}| \leq r_k \leq \min \left\{ 1; \bar{q}_{ev} \max_{i,j} |A_{t-ij}^{-1}| \right\} \quad (\text{for } A_{t-ij}^{-1} \neq 0) \quad (5.43)$$

Thus, eqns. (5.41) and (5.42) give $\underline{q}_{ev}$ and $\bar{q}_{ev}$. Then $\underline{r}$ and $\bar{r}$ can be computed by eqn.

(5.43). Substituting them into eqn. (5.40), the GST bounds can be obtained.

For the RC-ladder, RC-tree, RLC-ladder and RLC-tree interconnect networks, the bounds of their GST are given in the following four sub-sections.

5.3.3 RC-Ladder Networks

Based on eqn. (5.8), the RC-ladder network in Figure 5.1 has

$$\mathbf{A}_t^{-1} = RC \begin{bmatrix} 1 & 1 & 1 & 1 & \cdots & 1 & 1(1+\alpha) \\ 1 & 2 & 2 & 2 & \cdots & 2 & 2(1+\alpha) \\ 1 & 2 & 3 & 3 & \cdots & 3 & 3(1+\alpha) \\ \vdots & \vdots & \vdots & & & \vdots & \vdots \\ 1 & 2 & 3 & \cdots & n-2 & n-1 & (n-1)(1+\alpha) \\ 1 & 2 & 3 & \cdots & n-2 & n-1 & n(1+\alpha) \end{bmatrix}_{n \times n} \quad (5.44)$$

Thus, eqns. (5.41) and (5.42) imply (Marinov and Shivakumar 1996)

$$\underline{q}_{ev} = \frac{1}{RC\beta} \quad \text{and} \quad \bar{q}_{ev} = \frac{1}{RC(n+\alpha)} \quad (5.45)$$

where

$$\beta = \frac{n(n-1)}{2} + n(1+\alpha) \quad (5.46)$$

Besides, eqn. (5.43) leads to

$$\underline{r} = \underline{q}_{ev} RC = 1/\beta \quad \text{and} \quad \bar{r} = \min \{1; \bar{q}_{ev} n(1+\alpha) RC\} = 1 \quad (5.47)$$

With eqns. (5.45), (5.46) and (5.47), the GST bounds of the RC-ladder network can be obtained from eqn. (5.39) so that

$$\underline{T}_\lambda = \max \{0; -RC(n + \alpha) \ln \lambda \beta\} \quad \text{and} \quad \bar{T}_\lambda = -RC\beta \ln(\lambda/\beta) \quad (5.48)$$

5.3.4 RC-Tree Networks

For the *RC*-tree network shown in Figure 5.2, we have

[illegible]

and all other entries of $\mathbf{A}_t^{-1}$ that is not shown in the above equation are equal to 1.

Since $n = 1 + \sum_{k=1}^m \chi_k$ (eqn. (5.2)), we can obtain from eqns. (5.41) and (5.42) that

$$\max \{V_{row_lower}, V_{column_lower}\} \leq \frac{1}{q_{ev}} \leq \min \{V_{row_upper}, V_{column_upper}\} \quad (5.50)$$

where

V_{row_lower} is the minimum of the sums of elements' values in all row of A_t^{-1} ,

V_{row_upper} is the maximum of the sums of elements' values in all row of A_t^{-1} ,

V_{column_lower} is the minimum of the sums of elements' values in all column of A_t^{-1} ,

V_{column_upper} is the maximum of the sums of elements' values in all column of A_t^{-1} .

Chapter 5: Delay Time Bounds for RC- and RLC-Cell Networks

Since all the elements of A_t^{-1} of the RC-tree network are positive, it can be derived from eqns. (5.41), (5.42) and (5.49) that

$$V_{row_lower} = \min_i \sum_{j=1}^n A_{t_ij}^{-1} = (n + \alpha)RC \quad (5.51)$$

$$V_{row_upper} = \max_i \sum_{j=1}^n A_{t_ij}^{-1} = \left[n + \frac{\chi_m(\chi_m + 1)}{2} + (\chi_m + 1)\alpha \right] RC \quad (5.52)$$

$$V_{column_lower} = \min_j \sum_{i=1}^n A_{t_ij}^{-1} = nRC \quad (5.53)$$

$$V_{column_upper} = \max_j \sum_{i=1}^n A_{t_ij}^{-1} = \left[n + \frac{\chi_m(\chi_m + 1)}{2} \right] (1 + \alpha) RC \quad (5.54)$$

Because $\alpha \geq 0$, it can easily be shown that

$$V_{row_lower} - V_{column_lower} = \alpha RC \geq 0 \quad (5.55)$$

$$V_{row_upper} - V_{column_upper} = - \left[n + \frac{(\chi_m + 1)(\chi_m - 2)}{2} \right] \cdot \alpha RC \leq 0 \quad (5.56)$$

Hence, eqn.(5.50) can be developed to

$$(n + \alpha)RC \leq \frac{1}{q_{ev}} \leq \left[n + \frac{\chi_m(\chi_m + 1)}{2} + (\chi_m + 1)\alpha \right] RC \quad (5.57)$$

Thus,

$$\underline{q}_{ev} = \frac{1}{RC\varpi} \quad \text{and} \quad \bar{q}_{ev} = \frac{1}{RC(n + \alpha)} \quad (5.58)$$

where

$$\varpi = n + \frac{\chi_m(\chi_m + 1)}{2} + (\chi_m + 1)\alpha \quad (5.59)$$

Also, eqn. (5.43) gives

$$\underline{r} = \underline{q}_{ev} RC = \frac{1}{\varpi} \quad \text{and} \quad \bar{r} = \min \{1; \bar{q}_{ev} n(1 + \alpha)RC\} = 1 \quad (5.60)$$

With eqns. (5.58), (5.59) and (5.60), we have from eqn. (5.39),

$$\underline{T}_\lambda = \max \{0; -RC(n + \alpha) \ln \lambda \varpi\} \quad \text{and} \quad \bar{T}_\lambda = -RC\varpi \ln \frac{\lambda}{\varpi} \quad (5.61)$$

5.3.5 RLC-Ladder Networks

For the RLC-ladder network shown in Figure 5.3, $\mathbf{A}_t^{-1}$ is a $2n \times 2n$ square matrix having the following expression:

$$\mathbf{A}_t^{-1} = \begin{bmatrix} 0 & -C & 0 & -C & 0 & -C & 0 & -C & \cdots & 0 & -C(1+\alpha) \\ L & RC & 0 & RC & 0 & RC & 0 & RC & \cdots & 0 & RC(1+\alpha) \\ 0 & 0 & 0 & -C & 0 & -C & 0 & -C & \cdots & 0 & -C(1+\alpha) \\ L & RC & L & RC \sum_{j=0}^1 u^j & 0 & RC \sum_{j=0}^1 u^j & 0 & RC \sum_{j=0}^1 u^j & \cdots & 0 & RC(1+\alpha) \sum_{j=0}^1 u^j \\ 0 & 0 & 0 & 0 & 0 & -C & 0 & -C & \cdots & 0 & -C(1+\alpha) \\ L & RC & L & RC \sum_{j=0}^1 u^j & L & RC \sum_{j=0}^2 u^j & 0 & RC \sum_{j=0}^2 u^j & \cdots & 0 & RC(1+\alpha) \sum_{j=0}^2 u^j \\ 0 & 0 & 0 & 0 & 0 & 0 & \ddots & \ddots & \vdots & \vdots & \vdots \\ L & RC & L & RC \sum_{j=0}^1 u^j & L & RC \sum_{j=0}^2 u^j & \ddots & 0 & -C & 0 & -C(1+\alpha) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & L & RC \sum_{j=0}^{n-2} u^j & 0 & RC(1+\alpha) \sum_{j=0}^{n-2} u^j \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & -C(1+\alpha) \\ L & RC & L & RC \sum_{j=0}^1 u^j & L & RC \sum_{j=0}^2 u^j & \cdots & L & RC \sum_{j=0}^{n-2} u^j & L & RC(1+\alpha) \sum_{j=0}^{n-1} u^j \end{bmatrix}_{2n \times 2n} \quad (5.62)$$

where u is a scaling factor shown in Figure 5.3.

Simplifying $\mathbf{A}_t^{-1}$, it is observed that we need to consider two situations: $u = 1$ and

$u \neq 1$. For networks with $u \neq 1$, the coefficient $\sum_{j=0}^k u^j$ can be described by

$$\sum_{j=0}^k u^j = \frac{u^{k+1} - 1}{u - 1} \quad (u \neq 1) \quad (5.63)$$

However, when $u = 1$, this formula lead to division by zero and are obviously not applicable. Hence, $u = 1$ is regarded as a special case, and the corresponding formula is

$$\sum_{j=0}^k u^j = k + 1 \quad (u = 1) \quad (5.64)$$

Thus, $\mathbf{A}_t^{-1}$ has two expressions for different u values.

Chapter 5: Delay Time Bounds for RC- and RLC-Cell Networks

As shown in eqn. (5.50), the bounds of q_{ev} are decided by V_{row_lower} , V_{column_lower} ,

V_{row_upper} and V_{column_upper} which are computed from eqns. (5.41), (5.42) and (5.62) as

$$\begin{aligned} V_{row_lower} &= \min_i \sum_{j=1}^{2n} |A_{t-ij}^{-1}| = \min \left\{ \sum_{j=1}^{2n} |A_{t-2j}^{-1}|, \sum_{j=1}^{2n} |A_{t-(2n-1)j}^{-1}| \right\} \\ &= \min \{L + RC(n + \alpha), C(1 + \alpha)\} \end{aligned} \quad (5.65)$$

$$\begin{aligned} V_{column_lower} &= \min_j \sum_{i=1}^{2n} |A_{t-ij}^{-1}| = \min \left\{ \sum_{i=1}^{2n} |A_{t-i2}^{-1}|, \sum_{i=1}^{2n} |A_{t-i(2n-1)}^{-1}| \right\} \\ &= \min \{(nR + 1)C, L\} \end{aligned} \quad (5.66)$$

$$\begin{aligned} V_{row_upper} &= \max_i \sum_{j=1}^{2n} |A_{t-ij}^{-1}| = \max \left\{ \sum_{j=1}^{2n} |A_{t-1j}^{-1}|, \sum_{j=1}^{2n} |A_{t-(2n)j}^{-1}| \right\} \\ &= \begin{cases} \max \left\{ (n + \alpha)C, nL + RC \left[\frac{n(n+1)}{2} + n\alpha \right] \right\} & (u = 1) \\ \max \left\{ (n + \alpha)C, nL + RC \left[\frac{\alpha(u^n - 1)}{u - 1} + \frac{u^{n+1} - (n+1)u + n}{(u-1)^2} \right] \right\} & (u \neq 1) \end{cases} \end{aligned} \quad (5.67)$$

$$\begin{aligned} V_{column_upper} &= \max_j \sum_{i=1}^{2n} |A_{t-ij}^{-1}| = \max \left\{ \sum_{i=1}^{2n} |A_{t-i1}^{-1}|, \sum_{i=1}^{2n} |A_{t-i(2n)}^{-1}| \right\} \\ &= \begin{cases} \max \left\{ nL, C(1 + \alpha) \left[n + R \cdot \frac{n(n+1)}{2} \right] \right\} & (u = 1) \\ \max \left\{ nL, C(1 + \alpha) \left[n + R \cdot \frac{u^{n+1} - (n+1)u + n}{(u-1)^2} \right] \right\} & (u \neq 1) \end{cases} \end{aligned} \quad (5.68)$$

Thus, from eqn. (5.50), the lower and upper bounds of q_{ev} are derived as

$$\underline{q}_{ev} = \frac{1}{\min \{V_{row_upper}, V_{column_upper}\}} \quad (5.69)$$

$$\bar{q}_{ev} = \frac{1}{\max \{V_{row_lower}, V_{column_lower}\}} \quad (5.70)$$

From eqn. (5.62), for $A_{t-ij}^{-1} \neq 0$, we have

$$\min_{i,j} |A_{t-ij}^{-1}| = \min \{C, L, RC\} \quad (5.71)$$

$$\begin{aligned} \max_{i,j} |A_{t-ij}^{-1}| &= \max \left\{ C, L, RC(1+\alpha) \sum_{j=0}^{n-1} u^j \right\} \\ &= \begin{cases} \max \{ C, L, RC(1+\alpha) \cdot n \} & (u = 1) \\ \max \left\{ C, L, RC(1+\alpha) \cdot \frac{u^n - 1}{u - 1} \right\} & (u \neq 1) \end{cases} \end{aligned} \quad (5.72)$$

The lower and upper bounds of r_k can be obtained from eqn. (5.43) as

$$\underline{r} = \underline{q}_{ev} \min \{ C, L, RC \} \quad (5.73)$$

$$\bar{r} = \begin{cases} \min \left\{ 1; \bar{q}_{ev} \max \{ C, L, RC(1+\alpha) \cdot n \} \right\} & (u = 1) \\ \min \left\{ 1; \bar{q}_{ev} \max \left\{ C, L, RC(1+\alpha) \cdot \frac{u^n - 1}{u - 1} \right\} \right\} & (u \neq 1) \end{cases} \quad (5.74)$$

Substituting the lower and upper bounds of r_k and q_{ev} into eqn. (5.40), the bounds of GST are found.

5.3.6 RLC-Tree Networks

For the RLC-tree network shown in Figure 5.4, based on A_t of eqn. (5.16), A_t^{-1} is derived as

$$\mathbf{A}_t^{-1} = \begin{bmatrix} 0 & -C & 0 & -C & \cdots & 0 & -C & C & -C \\ L & RC & 0 & RC & \cdots & 0 & RC & 0 & RC \\ 0 & 0 & & & & & & & \\ L & RC & & & & & & & \\ \vdots & \vdots & & & & & & & \\ 0 & 0 & & & & & & & \\ L & RC & & & & & & & \\ 0 & 0 & & & & & & & \\ L & RC & & & & & & & \end{bmatrix}_{2n \times 2n} \quad \mathbf{H}_m \quad \begin{bmatrix} 0 & & & \\ & \ddots & & \\ & & 0 & \\ & & & 1+\alpha \end{bmatrix}_{2n \times 2n} \quad (5.75)$$

where $\mathbf{H}_k$ is a square matrix whose order is

Chapter 5: Delay Time Bounds for RC- and RLC-Cell Networks

$$\begin{cases} N_{H_{-k}} = 2n - 2 - 2 \sum_{i=k+1}^m \chi_i & k \in [1, m-1] \\ N_{H_{-m}} = 2n - 2 & k = m \end{cases} \quad (5.76)$$

and

$$\begin{cases} \mathbf{H}_1 = \mathbf{W}_k & k = 1 \\ \mathbf{H}_k = \begin{bmatrix} \mathbf{H}_{k-1} & \mathbf{U}_k \\ \mathbf{V}_k & \mathbf{W}_k \end{bmatrix} & k \in [2, m] \end{cases} \quad (5.77)$$

In eqn. (5.77), when $k \in [2, m]$, $\mathbf{U}_k$ is a $N_{H_{-(k-1)}} \times 2\chi_k$ matrix and $\mathbf{V}_k$ is the transpose matrix of $\mathbf{U}_k$ where

$$\mathbf{U}_k = RC \begin{bmatrix} 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & 1 & \cdots & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & 1 & \cdots & 0 & 1 \end{bmatrix}_{N_{H_{-(k-1)}} \times 2\chi_k} \quad (5.78)$$

$$\text{and } \mathbf{V}_k = \mathbf{U}_k^T \quad (5.79)$$

As well, for $k \in [1, m]$, $\mathbf{W}_k$ is a $2\chi_k \times 2\chi_k$ square matrix. It is given by

$$\mathbf{W}_k = \begin{bmatrix} 0 & -C & 0 & -C & 0 & -C & 0 & \cdots & -C \\ L & RC \sum_{j=0}^1 u^j & 0 & RC \sum_{j=0}^1 u^j & 0 & RC \sum_{j=0}^1 u^j & 0 & \cdots & RC \sum_{j=0}^1 u^j \\ 0 & 0 & 0 & -C & 0 & -C & 0 & \cdots & -C \\ L & RC \sum_{j=0}^1 u^j & L & RC \sum_{j=0}^2 u^j & 0 & RC \sum_{j=0}^2 u^j & 0 & \cdots & RC \sum_{j=0}^2 u^j \\ 0 & 0 & 0 & 0 & 0 & -C & 0 & \cdots & -C \\ L & RC \sum_{j=0}^1 u^j & L & RC \sum_{j=0}^2 u^j & L & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & RC \sum_{j=0}^{\chi_k-1} u^j & 0 & RC \sum_{j=0}^{\chi_k-1} u^j \\ \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & 0 & 0 & -C \\ L & RC \sum_{j=0}^1 u^j & L & RC \sum_{j=0}^2 u^j & L & \cdots & RC \sum_{j=0}^{\chi_k-1} u^j & L & RC \sum_{j=0}^{\chi_k} u^j \end{bmatrix} \quad (5.80)$$

For example, for a 7-cell RLC-tree network with 3 branches: $\chi_1 = 1$, $\chi_2 = 2$, $\chi_3 = 3$ and

eqns. (5.74)-(5.78) give $\mathbf{A}_t^{-1}$ as

Chapter 5: Delay Time Bounds for RC- and RLC-Cell Networks

$$\mathbf{A}_t^{-1} = \begin{bmatrix} 0 & -C & 0 & -C & 0 & -C & 0 & -C & 0 & -C & 0 & -C & 0 & -C(1+\alpha) \\ L & RC & 0 & RC & 0 & RC & 0 & RC & 0 & RC & 0 & RC & 0 & RC(1+\alpha) \\ 0 & 0 & 0 & -C & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ L & RC & L & RC \sum_{j=0}^1 u^j & 0 & RC & 0 & RC & 0 & RC & 0 & RC & 0 & RC(1+\alpha) \\ 0 & 0 & 0 & 0 & 0 & -C & 0 & -C & 0 & 0 & 0 & 0 & 0 & 0 \\ L & RC & 0 & RC & L & RC \sum_{j=0}^1 u^j & 0 & RC \sum_{j=0}^1 u^j & 0 & RC & 0 & RC & 0 & RC(1+\alpha) \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -C & 0 & 0 & 0 & 0 & 0 & 0 \\ L & RC & 0 & RC & L & RC \sum_{j=0}^1 u^j & L & RC \sum_{j=0}^2 u^j & 0 & RC & 0 & RC & 0 & RC(1+\alpha) \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -C & 0 & -C & 0 & -C(1+\alpha) \\ L & RC & 0 & RC & 0 & RC & 0 & RC & L & RC \sum_{j=0}^1 u^j & 0 & RC \sum_{j=0}^1 u^j & 0 & RC(1+\alpha) \sum_{j=0}^1 u^j \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -C & 0 & -C(1+\alpha) \\ L & RC & 0 & RC & 0 & RC & 0 & RC & L & RC \sum_{j=0}^1 u^j & L & RC \sum_{j=0}^2 u^j & 0 & (1+\alpha) RC \sum_{j=0}^2 u^j \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -C(1+\alpha) \\ L & RC & 0 & RC & 0 & RC & 0 & RC & L & RC \sum_{j=0}^1 u^j & L & RC \sum_{j=0}^2 u^j & L & (1+\alpha) RC \sum_{j=0}^3 u^j \end{bmatrix}_{14 \times 14}$$

With eqn. (5.50), it is derived from eqns. (5.41), (5.42) and (5.75) that

$$V_{row_lower} = \min_i \sum_{j=1}^{2n} |A_{t_ij}^{-1}| = \min \left\{ \sum_{j=1}^{2n} |A_{t_2j}^{-1}|, \sum_{j=1}^{2n} |A_{t_-(2\chi_1+1)j}^{-1}| \right\} \quad (5.81)$$

$$= \min \{L + RC(n + \alpha), C\},$$

$$V_{column_lower} = \min_j \sum_{i=1}^{2n} |A_{t_ij}^{-1}| = \min \left\{ \sum_{i=1}^{2n} |A_{t_i2}^{-1}|, \sum_{i=1}^{2n} |A_{t_i(2n-1)}^{-1}| \right\} \quad (5.82)$$

$$= \min \{(nR + 1)C, L\},$$

$$V_{row_upper} = \max_i \sum_{j=1}^{2n} |A_{t_ij}^{-1}| = \max \left\{ \sum_{j=1}^{2n} |A_{t_1j}^{-1}|, \sum_{j=1}^{2n} |A_{t_-(2n)j}^{-1}| \right\} \quad (5.83)$$

$$= \begin{cases} \max \left\{ (n + \alpha)C, \right. \\ \left. (\chi_m + 1)L + RC \left[n - \chi_m + \frac{(\chi_m + 3)\chi_m}{2} + (\chi_m + 1)\alpha \right] \right\} & (u = 1) \\ \max \left\{ (n + \alpha)C, \right. \\ \left. (\chi_m + 1)L + RC \left[n - \chi_m + \frac{u^2(u^{\chi_m} - 1) + (1 - u)\chi_m}{(u - 1)^2} + \frac{(u^{\chi_m + 1} - 1)\alpha}{u - 1} \right] \right\} & (u \neq 1) \end{cases}$$

Chapter 5: Delay Time Bounds for RC- and RLC-Cell Networks

$$V_{column_upper} = \max_j \sum_{i=1}^{2n} |A_{t_ij}^{-1}| = \max \left\{ \sum_{i=1}^{2n} |A_{t_i1}^{-1}|, \sum_{i=1}^{2n} |A_{t_i(2n)}^{-1}| \right\} \quad (5.84)$$

$$= \begin{cases} \max \left\{ nL, C(1+\alpha) \left\{ \chi_m + 1 + R \left[n - \chi_m + \frac{(\chi_m + 3)\chi_m}{2} \right] \right\} \right\} & (u = 1) \\ \max \left\{ nL, C(1+\alpha) \left\{ \chi_m + 1 + R \left[n - \chi_m + \frac{u^2(u^{\chi_m} - 1) + (1-u)\chi_m}{(u-1)^2} \right] \right\} \right\} & (u \neq 1) \end{cases}$$

Recalling eqns. (5.69) and (5.70), the lower and upper bounds of q_{ev} are given by

$$\underline{q}_{ev} = \frac{1}{\min\{V_{row_upper}, V_{column_upper}\}} \quad \text{and} \quad \bar{q}_{ev} = \frac{1}{\max\{V_{row_lower}, V_{column_lower}\}}$$

For $A_{t_ij}^{-1} \neq 0$, eqn. (5.43) gives

$$\underline{r} = \underline{q}_{ev} \min_{i,j} |A_{t_ij}^{-1}| \quad \text{and} \quad \bar{r} = \min \left\{ 1; \bar{q}_{ev} \max_{i,j} |A_{t_ij}^{-1}| \right\} \quad (5.85)$$

Considering $\chi_m \geq \chi_{m-1} \geq \dots \geq \chi_1$ and eqns. (5.75)-(5.80), the following relationships exist

$$\min_{i,j} |A_{t_ij}^{-1}| = \min\{C, L, RC\} \quad (5.86)$$

$$\begin{aligned} \max_{i,j} |A_{t_ij}^{-1}| &= \max \left\{ C, L, RC(1+\alpha) \sum_{j=0}^{\chi_m} u^j \right\} \\ &= \begin{cases} \max \{ C, L, RC(\alpha+1)(\chi_m+1) \} & (u = 1) \\ \max \left\{ C, L, RC(\alpha+1) \cdot \frac{u^{\chi_m+1} - 1}{u - 1} \right\} & (u \neq 1) \end{cases} \end{aligned} \quad (5.87)$$

Then the lower and upper bounds of r_k can be calculated by eqn. (5.85) as

$$\underline{r} = \underline{q}_{ev} \min\{C, L, RC\} \quad (5.88)$$

$$\bar{r} = \begin{cases} \min \left\{ 1; \bar{q}_{ev} \max \{ C, L, RC(\alpha+1)(\chi_m+1) \} \right\} & (u = 1) \\ \min \left\{ 1; \bar{q}_{ev} \max \left\{ C, L, RC(\alpha+1) \cdot \frac{u^{\chi_m+1} - 1}{u - 1} \right\} \right\} & (u \neq 1) \end{cases} \quad (5.89)$$

Finally, based on eqn. (5.40), the GST bounds can be computed by substituting

$\underline{q}_{ev}$, $\bar{q}_{ev}$, $\underline{r}$ and $\bar{r}$.

5.4 Delay Time Bounds of Interconnect Networks

To convert GST to delay time, it is necessary to perform the following. For the networks considered in this chapter, at $t = T_D$, there exists

$$v_i(T_D) = v_n(T_D) = v_n^\infty / 2 \quad (5.90)$$

where $v_n(t)$ is the transient voltage of the n^{th} node, and v_n^∞ is the steady value of $v_n(t)$ when $t \rightarrow \infty$. Since v_n^∞ can reach v_s in the steady status, $v_i(T_D)$ can be computed and equals $v_s/2$.

Furthermore, from eqns. (5.24) and (5.26),

$$\lambda = f(t) = \frac{\|\mathbf{z}(t)\|}{\|\mathbf{z}^0\|} \quad (5.91)$$

where $\mathbf{z}(t)$ and $\mathbf{z}^0$ have been defined in eqn (5.23) as

$$\begin{cases} \mathbf{z}(t) = \mathbf{x}(t) - \mathbf{x}^\infty \\ \mathbf{z}^0 = \mathbf{x}^0 - \mathbf{x}^\infty \end{cases} \quad (5.23)$$

and $\|\mathbf{z}(t)\|$ has been presented in eqn. (5.26). In eqn. (5.23), for RC-cell and RLC-cell networks, $\mathbf{x}(t)$ has been expressed by eqns. (5.6) and (5.11), respectively. From eqn. (5.88), at the delay time, the corresponding λ value can be defined as,

$$\lambda_D = f(T_D) = \frac{\|\mathbf{z}(T_D)\|}{\|\mathbf{z}^0\|} \quad (5.92)$$

In this way, the delay of the network can be considered as a special case of the global switching time,

$$T_D = T_\lambda \quad \text{when } \lambda = \lambda_D \quad (5.93)$$

In other words, the delay is equal to the GST as long as $\lambda = \lambda_D$. Considering $v_i(T_D) = v_s/2$, at $t = T_D$, the state elements of a network such as inductor currents and capacitor voltages can be calculated by using basic circuit theory such as Kirchhoff's Law (Carlson 2000). Hence, T_D for RC- and RLC-cell networks can be computed.

Chapter 5: Delay Time Bounds for RC- and RLC-Cell Networks

Combining eqns. (5.93) and (5.40), the delay time bounds can be represented by

$$\underline{T}_D = \max \left\{ 0; -\frac{1}{\underline{q}_{ev}} \ln \frac{\lambda_D \bar{r}}{\underline{r}} \right\} \quad \text{and} \quad \bar{T}_D = -\frac{1}{\underline{q}_{ev}} \ln \frac{\lambda_D \underline{r}}{\bar{r}} \quad (5.94)$$

where the equations to compute the values of $\underline{q}_{ev}$, $\bar{q}_{ev}$, $\underline{r}$, $\bar{r}$ of different networks have been given in Sections 5.3.3-5.3.6.

5.5 Numerical Examples and Analysis of Delay Time Bounds

5.5.1 RC-Ladder Networks

Assuming that $RC = 1$, the RC-ladder networks with different cell number n are selected as examples. Each case has three subcases corresponding to three different α values (different loads).

Their delay bounds are obtained and compared with the exact delays in Tables 5.1 and 5.2. In these examples, the delay time bound width ε_D is defined by

$$\varepsilon_D = (\bar{T}_D - \underline{T}_D) / T_D \quad (5.95)$$

where $\underline{T}_D$ is the lower delay bound, $\bar{T}_D$ is the upper delay bound and T_D is the exact delay.

5.5.2 RC-Tree Networks

Assume that $RC = 1$. The RC-tree networks in Figure 5.2 with a series of different cell number n , branch number m and branch cell number χ_k are selected as examples. Each case includes three subcases corresponding to three different α values (different loads).

To the networks with $n = 5, 10$ and 15 , $m = 2$ and $\chi_1 = 2$, the corresponding values of χ_2 are chosen as 2, 7 and 12. Their delay bounds are obtained and compared

Chapter 5: Delay Time Bounds for RC- and RLC-Cell Networks

with the exact delays in Tables 5.3, 5.4 and 5.5. Also, to the networks with $n = 6, 10$ and 15 , $m = 4$ and different values of χ_k , the comparison results are listed in Tables 5.6, 5.7 and 5.8.

Table 5.1 Delay bounds for RC-ladder networks with $n = 20$

$n = 20$					
α	λ_D	$\underline{T}_D$ (s)	T_D (s)	$\overline{T}_D$ (s)	ε_D
0	0.32492435	8.17437095	159.03710953	329.14700485	2.0182
20	0.27878655	0	442.33516215	1609.93052379	3.6396
40	0.27166535	0	720.07598359	3261.14488442	4.5289

Table 5.2 Delay bounds for RC-ladder networks with $n = 30$

$n = 30$					
α	λ_D	$\underline{T}_D$ (s)	T_D (s)	$\overline{T}_D$ (s)	ε_D
0	0.32372540	17.47075680	352.19859075	735.27032941	2.0381
30	0.27475617	0	990.04550321	3628.86235558	3.6653
60	0.26754170	0	1615.05185185	7364.29938506	4.5598

Table 5.3 Delay bounds for RC-tree networks with $n = 5$, $m = 2$ and $\chi_m = 2$

$n = 5$, $m = 2$, $\chi_1 = 2$ and $\chi_2 = 2$					
α	λ_D	$\underline{T}_D$ (s)	T_D (s)	$\overline{T}_D$ (s)	ε_D
0	0.4323422	0	5.82375081	23.34383533	4.0084
5	0.3016980	0	16.85771134	99.67792849	5.9129
10	0.2833046	0	27.25658435	186.15511420	6.8297

Table 5.4 Delay bounds for RC-tree networks with $n = 10$, $m = 2$ and $\chi_m = 7$

$n = 10$, $m = 2$, $\chi_1 = 2$ and $\chi_2 = 7$					
α	λ_D	$\underline{T}_D$ (s)	T_D (s)	$\overline{T}_D$ (s)	ε_D
0	0.2957684	0	29.26963187	184.51905948	6.3041
10	0.2503799	0	85.52815794	726.34434360	8.4925
20	0.2446063	0	141.00595490	1325.88172142	9.4030

Table 5.5 Delay bounds for RC-tree networks with $n = 15$, $m = 2$ and $\chi_m = 12$

$n = 15, m = 2, \chi_1 = 2 \text{ and } \chi_2 = 12$					
α	λ_D	$\underline{T}_D(s)$	$T_D(s)$	$\bar{T}_D(s)$	ε_D
0	0.2955632	0	70.92965792	534.88690375	7.5411
10	0.2517372	0	208.51856420	2028.19106100	9.7267
30	0.2458476	0	343.84237890	3662.61802872	10.6520

Table 5.6 Delay bounds for RC-tree networks with $n = 6$, $m = 4$ and $\chi_m = 2$

$n = 6, m = 4, \chi_1 = 1, \chi_2 = 1, \chi_3 = 1 \text{ and } \chi_4 = 2$					
α	λ_D	$\underline{T}_D(s)$	$T_D(s)$	$\bar{T}_D(s)$	ε_D
0	0.3909152	0	6.84542141	28.22840280	4.1237
6	0.2768295	0	19.84743828	123.66513950	6.2308
12	0.2638137	0	32.29789030	231.26285684	7.1603

Table 5.7 Delay bounds for RC-tree networks with $n = 10$, $m = 4$ and $\chi_m = 3$

$n = 10, m = 4, \chi_1 = 1, \chi_2 = 2, \chi_3 = 3 \text{ and } \chi_4 = 3$					
α	λ_D	$\underline{T}_D(s)$	$T_D(s)$	$\bar{T}_D(s)$	ε_D
0	0.4074241	0	11.94072540	58.72782950	4.9183
10	0.2283616	0	41.31698008	308.12189140	7.4575
20	0.2127227	0	68.95048243	586.76294696	8.5099

Table 5.8 Delay bounds for RC-tree networks with $n = 15$, $m = 4$ and $\chi_m = 5$

$n = 15, m = 4, \chi_1 = 2, \chi_2 = 3, \chi_3 = 4 \text{ and } \chi_4 = 5$					
α	λ_D	$\underline{T}_D(s)$	$T_D(s)$	$\bar{T}_D(s)$	ε_D
0	0.4275736	0	18.24719106	101.71261666	5.5742
15	0.1958094	0	73.71016953	623.57837279	8.4599
30	0.1957963	0	125.51580610	1189.20662347	9.4746

5.5.3 RLC-Ladder Networks

The RLC-ladder networks (Figure 5.3) with diverse RLC cells are selected as examples. Each case has four subclasses corresponding to four different α values (different loads). Corresponding to two cases: $u=1$ and $u \neq 1$, the related values of R , L and C are set respectively as:

when $u=1$: $R=1k\Omega$, $L=1\mu H$ and $C=10nF$;

when $u \neq 1$: $R=1\Omega$, $L=1nH$ and $C=10pF$.

For all the examples, the delay bounds are compared with the exact delays as shown in Tables 5.9-5.14.

5.5.4 RLC-Tree Networks

The n -cell RLC-tree networks (Figure 5.4) with a series of diverse branch structures and u values are selected as examples. Each case has four subclasses corresponding to four different α values. Corresponding to two cases: $u=1$ or $u \neq 1$, the related values of R , L and C are set as the same as those used in RLC-ladder networks in Section 5.5.3.

The delay bounds are computed and compared with the exact delay in Tables 5.15-5.20.

Table 5.9 Delay bounds for RLC-ladder networks with $u=1$ and $n=20$

$u=1, n=20$					
α	λ_D	$\underline{T}_D$ (s)	T_D (s)	$\bar{T}_D$ (s)	ε_D
0	0.32643089	0	0.00159033	0.02808911	17.6624
20	0.27879074	0	0.00442326	0.08936290	20.2029
40	0.27166201	0	0.00720089	0.15312203	21.2643
80	0.26737404	0	0.01274817	0.28501036	22.3570

Table 5.10 Delay bounds for RLC-ladder networks with $u = 1$ and $n = 25$

$u = 1, n = 25$					
α	λ_D	$\underline{T}_D(s)$	$T_D(s)$	$\overline{T}_D(s)$	ε_D
0	0.32480937	0	0.00246143	0.04490602	18.2439
25	0.27636927	0	0.00688979	0.14337852	20.8103
50	0.26919053	0	0.01122994	0.24583267	21.8908
100	0.26488422	0	0.01989761	0.45758467	22.9970

Table 5.11 Delay bounds for RLC-ladder networks with $u = 5$ and $n = 13$

$u = 5, n = 13$					
α	λ_D	$\underline{T}_D(s)$	$T_D(s)$	$\overline{T}_D(s)$	ε_D
0	0.05025845	0	0.00287073	0.08678482	30.2309
13	0.04824336	0	0.03034057	1.09695864	36.1548
26	0.04813904	0	0.05783792	2.15178518	37.2037
52	0.04810850	0	0.11283456	4.31383746	38.2315

Table 5.12 Delay bounds for RLC-ladder networks with $u = 5$ and $n = 15$

$u = 5, n = 15$					
α	λ_D	$\underline{T}_D(s)$	$T_D(s)$	$\overline{T}_D(s)$	ε_D
0	0.04355743	0	0.07176837	2.49024323	34.6983
15	0.04175561	0	0.86427087	35.60550279	41.1972
30	0.04171352	0	1.65747269	70.03360719	42.2532
60	0.04169175	0	3.24392186	140.41298236	43.2849

Table 5.13 Delay bounds for RLC-ladder networks with $u = 10$ and $n = 10$

$u = 10, n = 10$					
α	λ_D	$\underline{T}_D(s)$	$T_D(s)$	$\overline{T}_D(s)$	ε_D
0	0.05613240	0	0.00894584	0.29400036	32.8645
10	0.05559620	0	0.08592036	3.22545859	37.5401
20	0.05557410	0	0.16293447	6.27902266	38.5371
40	0.05556200	0	0.31696380	12.53211066	39.5380

Table 5.14 Delay bounds for RLC-ladder networks with $u = 10$ and $n = 12$

$u = 10, n = 12$					
α	λ_D	$\underline{T}_D$ (s)	T_D (s)	$\overline{T}_D$ (s)	ε_D
0	0.04677708	0	0.89457384	35.31051719	39.4719
12	0.04632400	0	10.13226129	452.76092625	44.6851
24	0.04627167	0	19.37434483	885.31544786	45.6952
48	0.04634933	0	37.85792308	1767.96891264	46.7001

Table 5.15 Delay bounds for RLC-tree networks with $u = 1$, $m = 2$ and $\chi_m = 20$

$u = 1, n = 25, m = 2, \chi_1 = 4, \chi_2 = 20$					
α	λ_D	$\underline{T}_D$ (s)	T_D (s)	$\overline{T}_D$ (s)	ε_D
0	0.28040975	0	0.00179076	0.03205436	17.8999
25	0.23631670	0	0.00549723	0.11421147	20.7762
50	0.23071954	0	0.00914051	0.19994397	21.8745
100	0.22744900	0	0.01642019	0.37733065	22.9797

Table 5.16 Delay bounds for RLC-tree networks with $u = 1$, $m = 3$ and $\chi_m = 20$

$u = 1, n = 30, m = 3, \chi_1 = 4, \chi_2 = 5, \chi_3 = 20$					
α	λ_D	$\underline{T}_D$ (s)	T_D (s)	$\overline{T}_D$ (s)	ε_D
0	0.24110422	0	0.00184396	0.03314932	17.9773
25	0.20157561	0	0.00554793	0.11623071	20.9503
50	0.19660372	0	0.00919058	0.20283831	22.0702
100	0.19370203	0	0.01646991	0.38194950	23.1908

Table 5.17 Delay bounds for RLC-tree networks with $u = 5$, $m = 2$ and $\chi_m = 14$

$u = 5, n = 20, m = 2, \chi_1 = 5, \chi_2 = 14$					
α	λ_D	$\underline{T}_D$ (s)	T_D (s)	$\overline{T}_D$ (s)	ε_D
0	0.03266807	0	0.07176837	2.51767873	35.0806
20	0.03130334	0	1.12866725	47.46305976	42.0523
40	0.03127756	0	2.18628725	94.22423778	43.0978
80	0.03126287	0	4.30156014	189.79820365	44.1231

Table 5.18 Delay bounds for RLC-tree networks with $u = 5$, $m = 3$ and $\chi_m = 14$

$u = 5, n = 30, m = 3, \chi_1 = 5, \chi_2 = 10, \chi_3 = 14$					
α	λ_D	$\underline{T}_D$ (s)	T_D (s)	$\bar{T}_D$ (s)	ε_D
0	0.02177871	0	0.07176836	2.55634692	35.6194
20	0.02086827	0	1.12866721	48.12046675	42.6348
40	0.02085018	0	2.18628723	95.50051788	43.6816
80	0.02084181	0	4.30156021	192.31166458	44.7074

Table 5.19 Delay bounds for RLC-tree networks with $u = 10$, $m = 2$ and $\chi_m = 9$

$u = 10, n = 15, m = 2, \chi_1 = 5, \chi_2 = 9$					
α	λ_D	$\underline{T}_D$ (s)	T_D (s)	$\bar{T}_D$ (s)	ε_D
0	0.03741624	0	0.00894566	0.29900787	33.4249
15	0.03705840	0	0.12442701	4.81604027	38.7057
30	0.03704806	0	0.23994846	9.52751214	39.7065
60	0.03704259	0	0.47099329	19.17327665	40.7082

Table 5.20 Delay bounds for RLC-tree networks with $u = 10$, $m = 3$ and $\chi_m = 9$

$u = 10, n = 22, m = 3, \chi_1 = 5, \chi_2 = 7, \chi_3 = 9$					
α	λ_D	$\underline{T}_D$ (s)	T_D (s)	$\bar{T}_D$ (s)	ε_D
0	0.02551106	0	0.00894564	0.30373618	33.9535
15	0.02526709	0	0.12442702	4.88460062	39.2568
30	0.02526005	0	0.23994845	9.65990435	40.2582
60	0.02525711	0	0.47099326	19.43331156	41.2603

5.5.5 Analysis of Results

The results show that reasonable delay bounds can be obtained from theoretical formulation. Observing the bound width ε_D in Tables 5.1-5.20, the delay time bounds turn wider with the increase of load (the value of α).

When the same load is given, the networks with smaller χ_m have tighter delay time

bounds. Furthermore, for two tree networks with the same load and the same largest branch cell number χ_m , the one with more complicated branch structure will degrade delay time bounds.

5.6 Summary

Early stages of VLSI circuit design require repeated simulations, so fast analyzers using so-called “timing” simulators are needed. The delay time, regarded as one of most important parameters of VLSI circuit, needs to be estimated quickly. For a tree-type RC- or RLC-cell lumped circuit modeling a VLSI configuration, an efficient way to analyze this problem is to obtain the upper and lower bounds of delay. These delay bounds can be used to: (1) bound the delay, given the signal threshold; (2) bound the signal voltage, given a delay time; (3) certify that a circuit is “fast enough” to match the design expectation, given both the maximum delay and the voltage threshold.

Marinov and Shivakumar (1996) developed methods to obtain the global switching time (GST) bounds for RC-ladder networks. Since the RC-ladder structure is only a basic topology in real application, the GST bound calculations are extended to a more widely used interconnect structure, i.e. the tree structure. A method for converting the GST bounds to delay time bounds has also been presented.

New and efficient formulas have been derived for fast estimation of the delay bounds for RC-tree, RLC-ladder and RLC-tree networks. Numerical examples are given to show that delay bounds can be obtained from the theoretical formulas. It is also obtained that the obtained delay time bounds become wider with the increase of load. Furthermore, when the same load is given, a smaller branch cell number in the largest branch leads to tighter delay time bounds. Comparing two tree networks with the same load and the same largest branch cell number, the one with more complicated branch structure degraded delay time bounds.

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

6.1 Introduction

In this chapter, a new algorithm is applied to obtain the reduced-order model of the state matrix of a linear circuit via the Lanczos process (LP) (Feldmann and Freund 1995). It can generate an arbitrary-order model with little numerical degradation. The performance of the network can be described with just a 2nd-order model. Using the Lanczos algorithm, improved delay bounds compared to other methods (Marinov and Shivakumar 1996, Hou *et al.* 2000) can be derived. The RC-ladder network shown in Figure 6.1 is chosen as the example. This network is same as that in Figure 5.1 and repeated here for convenience. Numerical examples are given for illustration and analysis.

Figure 6.1 An n -cell RC-ladder network with capacitive load (same as Figure 5.1 and repeated for convenience)

6.2 Optimized Algorithm via Lanczos Process to Compute Global Switching Time Bounds of RC-Ladder Networks

6.2.1 System Reduction of Interconnect Networks

As presented by eqn. (3.1) (Chapter 3), a two-port interconnect network can be described by the state equations. We are interested in determining the step-response of the interconnect network with zero initial-conditions, which, in turn, can be used to determine

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

the response to any excitation. Considering zero initial-conditions, the direct coupling term d can be computed separately as zero; therefore the term $du(t)$ can be ignored (Pillage and Rohrer 1994). Then using any circuit-equation formulation method such as the modified nodal analysis, sparse tableau, etc. (Vlach and Singhal 1983), the two-port interconnect network can be described by the following system in another form (Feldmann and Freund 1995):

$$\begin{aligned} \mathbf{F}\dot{\mathbf{x}}(t) &= -\mathbf{G}\mathbf{x}(t) + \mathbf{h}u(t) \\ y(t) &= \mathbf{l}^T \mathbf{x}(t) \end{aligned} \quad (6.1)$$

where the vector $\mathbf{x}(t)$ represents the system state variables, the matrix $\mathbf{G}$ represents the contribution of memoryless elements, such as resistors, $\mathbf{F}$ represents the contribution from memory elements, such as capacitors and inductors, $y(t)$ is the output of interest, and the term $\mathbf{h}u(t)$ represents excitations from independent sources. Furthermore, comparing eqn. (3.1) with eqn. (6.1), it is obvious that

$$\mathbf{A} = -\mathbf{F}^{-1}\mathbf{G}, \quad \mathbf{b} = \mathbf{F}^{-1}\mathbf{h} \quad \text{and} \quad \mathbf{c} = \mathbf{l} \quad (6.2)$$

Applying Laplace transform to eqn. (6.1), its s -domain form is obtained as

$$\begin{aligned} s\mathbf{F}\mathbf{X}(s) &= -\mathbf{G}\mathbf{X}(s) + \mathbf{h}U(s) \\ Y(s) &= \mathbf{l}^T \mathbf{X}(s) \end{aligned} \quad (6.3)$$

where $\mathbf{X}(s)$, $U(s)$ and $Y(s)$ denote the Laplace transforms of $\mathbf{x}(t)$, $u(t)$ and $y(t)$, respectively. Defining the transfer function of the network as $H(s) = Y(s)/U(s)$, $H(s)$ is given by

$$H(s) = \mathbf{l}^T (\mathbf{G} + s\mathbf{F})^{-1} \mathbf{h} \quad (6.4)$$

Let s_0 be an arbitrary, but fixed expansion point such that the matrix $\mathbf{G} + s_0\mathbf{F}$ is nonsingular (Feldmann and Freund 1995). Using change of variables $s = s_0 + \sigma$, set

$$\mathbf{A}_L = -(\mathbf{G} + s_0\mathbf{F})^{-1}\mathbf{F} \quad \text{and} \quad \mathbf{r}_L = (\mathbf{G} + s_0\mathbf{F})^{-1}\mathbf{h} \quad (6.5)$$

Usually, the Maclaurin expansion with expansion point $s_0 = 0$ is used. Therefore, it is developed from eqn. (6.5) that

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

$$\mathbf{A}_L = -\mathbf{G}^{-1}\mathbf{F} \quad \text{and} \quad \mathbf{r}_L = \mathbf{G}^{-1}\mathbf{h} \quad (6.6)$$

Here, $\mathbf{A}_L$ and $\mathbf{r}_L$ are two important parameters which will be used in the following Lanczos algorithm.

6.2.2 Delay Bound Connection of Lanczos Process

In Section 5.2.1 (Chapter 5), we defined that

$$\mathbf{A}_t = -\mathbf{A} \quad (6.7)$$

In Section 5.3.2, the positive eigenvalue of $\mathbf{A}_t^T$ is denoted by q_{ev} and its corresponding eigenvector is $\mathbf{r}$ whose component is r_k . They have the following relationship (Lancaster and Tismenersky 1988):

$$\mathbf{A}_t^T \mathbf{r} = q_{ev} \mathbf{r} \quad (6.8)$$

Eqns. (6.2) and (6.6) give

$$\mathbf{A} = \mathbf{A}_L^{-1} \quad (6.9)$$

Thus, from eqns. (6.7) and (6.9), the matrix $\mathbf{A}_t^T$ corresponding to q_{ev} and $\mathbf{r}$ is expressed in terms of $\mathbf{A}_L$ as follows:

$$\mathbf{A}_t^T = (-\mathbf{A}_L^{-1})^T \quad (6.10)$$

Hence, the goal now is to use a reduced-order model to approximate $\mathbf{A}_t^T$ via the Lanczos process (LP) and compute the new lower and upper bounds of q_{ev} and r_k , denoted by $\underline{q}_{ev_LP}$, $\bar{q}_{ev_LP}$, $\underline{r}_{LP}$ and $\bar{r}_{LP}$ such that

$$\underline{q}_{ev_LP} \leq q_{ev} \leq \bar{q}_{ev_LP} \quad \text{and} \quad \underline{r}_{LP} \leq r_k \leq \bar{r}_{LP} \quad (6.11)$$

6.2.3 Reduced-Order Approximation via Lanczos Process

The Lanczos algorithm (Feldmann and Freund 1995) is an iterative procedure for the successive reduction of the square matrix $\mathbf{A}_L$ to a sequence of tridiagonal matrix $\mathbf{T}_q$. The

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

classic Lanczos algorithm and some of its key properties are recalled as follows:

Algorithm 6.1: Lanczos algorithm (Feldmann and Freund 1995):

Step (0): Initialize:

$$\text{set } \rho_1 = \|\mathbf{r}_L\|_2, \eta_1 = \|\mathbf{l}\|_2, \mathbf{v}_1 = \mathbf{r}_L / \rho_1 \text{ and } \mathbf{w}_1 = \mathbf{l} / \eta_1 \quad (6.12)$$

$$\text{set } \mathbf{v}_0 = \mathbf{w}_0 = 0 \text{ and } \delta_0 = 1.$$

For $k = 1, 2, \dots, q$:

$$\text{Step (1): Compute } \delta_k = \mathbf{w}_k^T \mathbf{v}_k \quad (6.13)$$

Step (2): Set

$$\alpha_k = \frac{\mathbf{w}_k^T \mathbf{A}_L \mathbf{v}_k}{\delta_k}, \beta_k = \frac{\delta_k}{\delta_{k-1}} \cdot \eta_k \text{ and } \gamma_k = \frac{\delta_k}{\delta_{k-1}} \cdot \rho_k \quad (6.14)$$

Step (3): Set

$$\mathbf{v} = \mathbf{A}_L \mathbf{v}_k - \mathbf{v}_k \alpha_k - \mathbf{v}_{k-1} \beta_k, \quad (6.15)$$

$$\mathbf{w} = \mathbf{A}_L^T \mathbf{w}_k - \mathbf{w}_k \alpha_k - \mathbf{w}_{k-1} \gamma_k. \quad (6.16)$$

Step (4): Set

$$\rho_{k+1} = \|\mathbf{v}\|_2, \eta_{k+1} = \|\mathbf{w}\|_2, \quad (6.17)$$

$$\mathbf{v}_{k+1} = \frac{\mathbf{v}}{\rho_{k+1}} \text{ and } \mathbf{w}_{k+1} = \frac{\mathbf{w}}{\eta_{k+1}}. \quad (6.18)$$

Exit.

Here, the parameters $\mathbf{l}$, $\mathbf{A}_L$ and $\mathbf{r}_L$ are the same as the definitions in eqns. (6.2) and (6.6). In the above algorithm, a division by 0 such as $\delta_k = 0$ in eqn. (6.14) will result in a breakdown. Furthermore, division by a nonzero yet small number $\delta_k \approx 0$ may lead to numerical instability (Feldmann and Freund 1995). However, the look-ahead Lanczos algorithm (Freund 1993, Celik *et al.* 2002) can be used to remedy this problem.

The tridiagonal matrix $\mathbf{T}_q$ generated by the Lanczos algorithm has the following properties:

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

$$\mathbf{T}_q = \begin{bmatrix} \alpha_1 & \beta_2 & 0 & \cdots & 0 \\ \rho_2 & \alpha_2 & \beta_3 & \ddots & \vdots \\ 0 & \rho_3 & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \beta_q \\ 0 & \cdots & 0 & \rho_q & \alpha_q \end{bmatrix}_{q \times q} \quad (6.19)$$

$\mathbf{T}_q$ is the best approximation to $\mathbf{A}_L$ even if q is much smaller than the order n of the $n \times n$ matrix $\mathbf{A}_L$. To the matrices with very large orders, the Lanczos algorithm still has good performance in obtaining the low-order approximation $\mathbf{T}_q$ where $q \ll n$.

In other words, with the value of k increasing, the parameters α_k , β_k and ρ_k have decreasing contribution to the performance of reduced-order approximation model $\mathbf{T}_q$. Considering this property of $\mathbf{T}_q$ and for computational convenience, a 2nd-order model ($q = 2$) is used to approximate the $n \times n$ square matrix $\mathbf{A}_L$.

6.2.4 Approximate Eigenvalue and Eigenvector of RC-Ladder Networks via Lanczos Process

To compute $\mathbf{T}_q$, the parameters in the Lanczos Algorithm, $\mathbf{l}$, $\mathbf{A}_L$ and $\mathbf{r}_L$, need to be obtained first. From eqns. (6.2) and (6.9),

$$\mathbf{l} = \mathbf{c} \text{ and } \mathbf{A}_L = \mathbf{A}^{-1} \quad (6.20)$$

Also, from eqns. (6.2) and (6.6), it can be derived that

$$\mathbf{r}_L = -\mathbf{A}_L \cdot \mathbf{b} \quad (6.21)$$

For the RC-ladder network in Figure 6.1, using the state-space method (Section 2.3.1), we have

$$\mathbf{A} = -\frac{1}{RC} \begin{bmatrix} & & & & 0 \\ & & & & 0 \\ & & & & \vdots \\ & & & & \vdots \\ & & & & 0 \\ & & & & -1 \\ 0 & 0 & \cdots & \cdots & 0 & \frac{-1}{1+\alpha} & \frac{1}{1+\alpha} \end{bmatrix}_{n \times n}, \quad \mathbf{b} = \frac{1}{RC} \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}_{n \times 1} \quad (6.22)$$

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

$$\mathbf{c}^T = [0 \ 0 \ \dots \ 0 \ 1]_{1 \times n} \quad \text{and} \quad d = 0 \quad (6.23)$$

where $\alpha = (C_t / C)$ and $\mathbf{J}$ has the form as

$$\mathbf{J} = \begin{bmatrix} 2 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & -1 & 2 & -1 & & \\ & & -1 & \ddots & & \\ & & & \ddots & -1 & \\ & & & -1 & 2 & -1 \\ & & & & -1 & 2 \end{bmatrix}_{(n-1) \times (n-1)} \quad (6.24)$$

From eqns. (6.20), (6.22) and (6.24), the formula of $\mathbf{A}_L$ is given as follows:

$$\mathbf{A}_L = RC \begin{bmatrix} -1 & -1 & -1 & -1 & \dots & -1 & -1(1+\alpha) \\ -1 & -2 & -2 & -2 & \dots & -2 & -2(1+\alpha) \\ -1 & -2 & -3 & -3 & \dots & -3 & -3(1+\alpha) \\ -1 & -2 & -3 & -4 & \dots & -4 & -4(1+\alpha) \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ -1 & -2 & -3 & -4 & \dots & -(n-1) & -(n-1)(1+\alpha) \\ -1 & -2 & -3 & -4 & \dots & -(n-1) & -n(1+\alpha) \end{bmatrix}_{n \times n} \quad (6.25)$$

Then combining eqns. (6.21), (6.22) and (6.25), $\mathbf{r}_L$ is given by

$$\mathbf{r}_L = [1 \ 1 \ \dots \ 1]_{1 \times n}^T \quad (6.26)$$

Following Algorithm 6.1 and considering eqns. (6.20)-(6.26), each step produces the results as shown as follows:

Step (0): Initialize.

$$\rho_1 = \|\mathbf{r}_L\|_2 = \sqrt{n}, \quad \eta_1 = \|\mathbf{1}\|_2 = 1, \quad (6.27)$$

$$\mathbf{v}_1 = \frac{\mathbf{r}_L}{\rho_1} = \frac{\sqrt{n}}{n} [1 \ 1 \ \dots \ 1]_{1 \times n}^T \quad \text{and} \quad \mathbf{w}_1 = \mathbf{1} / \eta_1 = [0 \ 0 \ \dots \ 0 \ 1]_{1 \times n}^T \quad (6.28)$$

For $k = 1$:

$$\text{Step (1): } \delta_1 = \mathbf{w}_1^T \mathbf{v}_1 = \sqrt{n}/n \quad (6.29)$$

$$\text{Step (2): } \alpha_1 = \frac{\mathbf{w}_1^T \mathbf{A}_L \mathbf{v}_1}{\delta_1} = RC \left[\frac{-(n+1)n}{2} - n\alpha \right], \quad (6.30)$$

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

$$\beta_1 = \frac{\delta_1}{\delta_0} \cdot \eta_1 = \frac{\sqrt{n}}{n} \quad \text{and} \quad \gamma_1 = \frac{\delta_1}{\delta_0} \cdot \rho_1 = 1. \quad (6.31)$$

$$\text{Step (3):} \quad \mathbf{v} = \mathbf{A}_L \mathbf{v}_1 - \mathbf{v}_1 \alpha_1 - \mathbf{v}_0 \beta_1 \quad (6.32)$$

$$\begin{aligned} &= \frac{RC\sqrt{n}}{n} \begin{bmatrix} -1(n+\alpha) + \frac{n(n+1)}{2} + n\alpha \\ -2(n+\alpha) + 1 + \frac{n(n+1)}{2} + n\alpha \\ \vdots \\ -j(n+\alpha) + \frac{j(j-1)}{2} + \frac{n(n+1)}{2} + n\alpha \\ \vdots \\ -(n-1)(n+\alpha) + \frac{(n-1)(n-2)}{2} + \frac{n(n+1)}{2} + n\alpha \\ 0 \end{bmatrix}_{n \times 1} \\ &\mathbf{w} = \mathbf{A}_L^T \mathbf{w}_1 - \mathbf{w}_1 \alpha_1 - \mathbf{w}_0 \gamma_1 = RC \begin{bmatrix} -1 \\ -2 \\ \vdots \\ -(n-1) \\ \frac{(n-1)n}{2} \end{bmatrix}_{n \times 1} \end{aligned} \quad (6.33)$$

$$\text{Step (4):} \quad \rho_2 = \|\mathbf{v}\|_2 = RC \sqrt{\frac{1}{n} \sum_{j=1}^n \left(-j(n+\alpha) + \frac{j(j-1)}{2} + \frac{n(n+1)}{2} + n\alpha \right)^2} \quad (6.34)$$

$$= \frac{\sqrt{15}}{30} RC \left[(n-1)(3n^3 + 15\alpha n^2 + 3n^2 + 5\alpha n - 2n + 20\alpha^2 n - 10\alpha^2 - 10\alpha - 2) \right]^{\frac{1}{2}}$$

$$\eta_2 = \|\mathbf{w}\|_2 = RC \sqrt{\left[\frac{(n-1)n}{2} \right]^2 + \sum_{j=1}^{n-1} (-j)^2} \quad (6.35)$$

$$= \frac{RC}{6} \cdot (9n^4 - 6n^3 - 9n^2 + 6n)^{1/2}$$

$$\mathbf{v}_2 = \frac{\mathbf{v}}{\rho_2} \quad \text{and} \quad \mathbf{w}_2 = \frac{\mathbf{w}}{\eta_2} \quad (6.36)$$

For $k = 2$:

$$\text{Step (5):} \quad \delta_2 = \mathbf{w}_2^T \mathbf{v}_2 = \frac{1}{\rho_2 \eta_2} \cdot \mathbf{w}^T \mathbf{v} \quad (6.37)$$

$$\begin{aligned} &= \frac{1}{\rho_2 \eta_2} \cdot \sum_{j=1}^{n-1} \frac{-j}{\sqrt{n}} \left[(n-j)\alpha + \frac{j(j-1) + n(n+1) - 2jn}{2} \right] \\ &= \frac{-1}{24\rho_2 \eta_2} \cdot n^{1/2} (n-1)(n+1)(n+2+4\alpha) \end{aligned}$$

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

$$\text{Step (6): } \alpha_2 = \frac{\mathbf{w}_2^T \mathbf{A}_L \mathbf{v}_2}{\delta_2} \quad (6.38)$$

$$= \frac{1}{\rho_2 \eta_2} \cdot \mathbf{w}^T \mathbf{A}_L \mathbf{v} = RC \cdot \frac{(n+2)(n-2)(n+3+6\alpha)}{30(n+2+4\alpha)}$$

$$\beta_2 = \frac{\delta_2}{\delta_1} \cdot \eta_2 \quad (6.39)$$

$$= \frac{-RC \cdot \sqrt{15} \cdot n(n+1)(n-1)(n+2+4\alpha)}{12 \left\{ (n-1) \cdot [3n^3 + (15\alpha+3)n^2 + (20\alpha^2+5\alpha-2)n - (10\alpha^2+10\alpha+2)] \right\}^{1/2}}$$

Step (7): Stop.

Since a 2nd-order $\mathbf{T}_q$ ($q=2$) is chosen to approximate $\mathbf{A}_L$, eqn. (6.19) gives that

$$\mathbf{T}_q(q=2) = \begin{bmatrix} \alpha_1 & \beta_2 \\ \rho_2 & \alpha_2 \end{bmatrix} \quad (6.40)$$

where the parameters α_1 , ρ_1 , α_2 and β_2 correspond to eqns. (6.30), (6.31), (6.38) and (6.39), respectively. Also, from eqn. (6.10), it can be derived that

$$\mathbf{A}_t^T = (-\mathbf{A}_L^{-1})^T \approx (-\mathbf{T}_q^{-1})^T = \hat{\mathbf{A}}_t^T \quad (6.41)$$

and

$$\hat{\mathbf{A}}_t^T = \begin{bmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{bmatrix} \quad (6.42)$$

where

$$d_{11} = \frac{-4(n^2-4)(n+6\alpha+3)}{RC \cdot n\zeta}, \quad d_{12} = \frac{4\sqrt{15}(n+4\alpha+2)\tau^{1/2}}{RC \cdot n\zeta}, \quad (6.43)$$

$$d_{21} = \frac{-10\sqrt{15}(n^2-1)(n+2+4\alpha)}{RC \cdot \tau^{1/2} \cdot \zeta}, \quad d_{22} = \frac{60(n+2\alpha+1)(n+4\alpha+2)}{RC \cdot \zeta} \quad (6.44)$$

with

$$\tau = 3n^4 + 15\alpha n^3 + (-5 - 10\alpha + 20\alpha^2)n^2 + (-15\alpha - 30\alpha^2)n + 10\alpha^2 + 10\alpha + 2, \quad (6.45)$$

$$\zeta = 3n^4 + (12 + 24\alpha)n^3 + (17 + 56\alpha + 56\alpha^2)n^2 + (12 + 24\alpha)n + (4 + 16\alpha + 16\alpha^2) \quad (6.46)$$

The eigenvalue ξ and the corresponding eigenvector $\mathbf{x}_\xi$ of the square matrix $\hat{\mathbf{A}}_t^T$

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

satisfy the following relation (Varga 1962, Lancaster and Tismenersky 1988):

$$\hat{\mathbf{A}}_t^T \mathbf{x}_\xi = \xi \mathbf{x}_\xi \quad (6.47)$$

So through solving the characteristic equation,

$$|\xi \cdot \mathbf{E} - \hat{\mathbf{A}}_t^T| = 0 \quad (6.48)$$

the eigenvalues of $\hat{\mathbf{A}}_t^T$ are given by

$$\xi_1 = \frac{2[14n^3 + (42 + 84\alpha)n^2 + (34 + 120\alpha + 120\alpha^2)n + 24\alpha + 12 - \psi^{1/2}]}{RC \cdot n\varsigma}, \quad (6.49)$$

$$\xi_2 = \frac{2[14n^3 + (42 + 84\alpha)n^2 + (34 + 120\alpha + 120\alpha^2)n + 24\alpha + 12 + \psi^{1/2}]}{RC \cdot n\varsigma} \quad (6.50)$$

where ς is given by eqn. (6.46) and

$$\psi = \quad (6.51)$$

$$\begin{aligned} &106n^6 + (636 + 1272\alpha)n^5 + (1486 + 5856\alpha + 5856\alpha^2)n^4 \\ &+ (1812 + 10344\alpha + 20160\alpha^2 + 13440\alpha^3)n^3 \\ &+ (1324 + 8832\alpha + 23232\alpha^2 + 28800\alpha^3 + 14400\alpha^4)n^2 \\ &+ (576 + 3072\alpha + 5760\alpha^2 + 3840\alpha^3)n + (576\alpha^2 + 576\alpha + 144) \end{aligned}$$

Given an eigenvalue of $\hat{\mathbf{A}}_t^T$, the eigenequation is

$$(\xi \mathbf{E} - \hat{\mathbf{A}}_t^T) \mathbf{x}_\xi = 0 \quad (6.52)$$

where ξ is the eigenvalue variable and can be substituted by ξ_1 or ξ_2 , i.e.,

$$\xi = \xi_1 \text{ or } \xi = \xi_2 \quad (6.53)$$

Solving eqn. (6.52) and considering eqns. (6.42), (6.43) and (6.44), the corresponding eigenvector $\mathbf{x}_\xi$ can be derived as

$$\mathbf{x}_\xi = \begin{bmatrix} x_1(\xi) \\ x_2(\xi) \end{bmatrix} = m_\xi \begin{bmatrix} 1 \\ (\xi - d_{11})/d_{12} \end{bmatrix} \quad (6.54)$$

where $x_k(\xi)$ are the two components of $\mathbf{x}_\xi$ ($k=1, 2$). Also, making $\|\mathbf{x}_\xi\|_2 = 1$ and noting that $d_{11} \leq 0$ (when $n \geq 2$) as well as $d_{12} \geq 0$, the value of m_ξ can be calculated as

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

$$m_{\xi} = \frac{d_{12}}{\sqrt{d_{12}^2 + (\xi - d_{11})^2}} \quad (6.55)$$

Then the eigenvector $\mathbf{x}_{\xi}$ can be developed from eqn. (6.53) as

$$\mathbf{x}_{\xi} = \begin{bmatrix} x_1(\xi) \\ x_2(\xi) \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{1 + \varphi_{\xi}^2}} \\ \frac{1}{\sqrt{1 + \varphi_{\xi}^{-2}}} \end{bmatrix} \quad \text{with} \quad \varphi_{\xi} = \frac{\xi - d_{11}}{d_{12}} \quad (6.56)$$

where d_{11} and d_{12} are given in eqn. (6.43).

6.2.5 Improvement Proof of Eigenvalue and Eigenvector via Lanczos Process

It will be proved that the GST bounds obtained from the new method via Lanczos process are tighter than those from the method given by Marinov and Shivakumar (1996). In Section 5.3.2, the formulations of GST bounds have been derived (same as eqn. (5.34) and recalled here for convenience):

$$\underline{T}_{\lambda} = \max \left\{ 0; -\frac{1}{\underline{q}_{ev}} \ln \frac{\lambda \bar{r}}{\underline{r}} \right\} \quad \text{and} \quad \bar{T}_{\lambda} = -\frac{1}{\underline{q}_{ev}} \ln \frac{\lambda \underline{r}}{\bar{r}} \quad (6.57)$$

For the RC-ladder network in Figure 6.1, Marinov and Shivakumar (1996) has presented:

$$\underline{q}_{ev} = \frac{1}{RC\beta} \quad \text{and} \quad \bar{q}_{ev} = \frac{1}{RC(n + \alpha)} \quad (6.58)$$

and

$$\underline{r} = \underline{q}_{ev} RC = 1/\beta \quad \text{and} \quad \bar{r} = \min \{1; \bar{q}_{ev} n(1 + \alpha) RC\} = 1 \quad (6.59)$$

where

$$\beta = \frac{n(n-1)}{2} + n(1 + \alpha) \quad (6.60)$$

More details are shown in Section 5.3.3.

The tighter GST bounds come from the tighter eigenvalue and eigenvector component bounds. Since the eigenvalue and eigenvector component bounds in eqns. (6.58) and (6.59) are known to be correct, these bounds can be used as the inspection criteria for the new

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

method via Lanczos process.

In the following, two eigenvalues of $\hat{\mathbf{A}}_i^T$ shown in eqns. (6.49) as well as (6.50) and their corresponding eigenvector components shown in eqn. (6.56) are discussed respectively. In addition, the n -cell RC-ladder network considered has $n \geq 2$.

(A) ξ_1 Improvement Proof

Theorem 6.1: The first eigenvalue of $\hat{\mathbf{A}}_i^T$, ξ_1 , lies within the eigenvalue bounds by Marinov and Shivakumar (1996), i.e.

$$\underline{q}_{ev} < \xi_1 < \bar{q}_{ev} \quad (6.61)$$

where ξ_1 is given by eqns. (6.49) and $\underline{q}_{ev}$ as well as $\bar{q}_{ev}$ are given by eqn. (6.58).

Theorem 6.1 can be expressed as

$$\text{Case 1: } \xi_1 - \underline{q}_{ev} > 0 \quad (6.62)$$

$$\text{Case 2: } \bar{q}_{ev} - \xi_1 > 0 \quad (6.63)$$

Proof of Case 1 of Theorem 6.1:

From eqns. (6.49), (6.58) and (6.62), it is derived that

$$\text{Gap1} = \xi_1 - \underline{q}_{ev} = \frac{2(V_{\text{Gap1_1}} - V_{\text{Gap1_2}})}{RCn(n+1+2\alpha)\zeta} \quad (6.64)$$

where ζ is defined in eqn. (6.46) and

$$\begin{aligned} V_{\text{Gap1_1}} &= 11n^4 + (44 + 88\alpha)n^3 + (59 + 232\alpha + 232\alpha^2)n^2 \\ &\quad + (34 + 188\alpha + 360\alpha^2 + 240\alpha^3)n + (8 + 32\alpha + 32\alpha^2) \\ V_{\text{Gap1_2}} &= (n + 1 + 2\alpha) \cdot \psi^{1/2} \end{aligned}$$

Because $n \geq 2$ and $\alpha \geq 0$, we have

$$\begin{aligned} \text{Gap2} &= V_{\text{Gap1_1}}^2 - V_{\text{Gap1_2}}^2 \\ &= 15n^8 + (120 + 452\alpha)n^7 + (370 + 2964\alpha + 4342\alpha^2)n^6 \\ &\quad + (520 + 7364\alpha + 22404\alpha^2 + 19688\alpha^3)n^5 \\ &\quad + (215 + 8636\alpha + 43378\alpha^2 + 78560\alpha^3 + 48928\alpha^4)n^4 \end{aligned} \quad (6.65)$$

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

$$\begin{aligned}
 &+(-320+4832\alpha+41388\alpha^2+116216\alpha^3+144960\alpha^4+69120\alpha^5)n^3 \\
 &+(-520+920\alpha+21700\alpha^2+78080\alpha^3+131776\alpha^4+115200\alpha^5+43200\alpha^6)n^2 \\
 &+(-320-480\alpha+5184\alpha^2+19328\alpha^3+24960\alpha^4+11520\alpha^5)n \\
 &+(-80-352\alpha-336\alpha^2+320\alpha^3+448\alpha^4)>0
 \end{aligned}$$

Also, since V_{Gap1_1} and $V_{Gap1_2} > 0$, it can be obtained from eqn. (6.65) that

$V_{Gap1_1} > V_{Gap1_2}$. Thus, based on eqn. (6.64), eqn. (6.62) can be proven. ■

Proof of Case 2 of Theorem 6.1:

From eqns. (6.49), (6.58) and (6.63), we have

$$Gap3 = \bar{q}_{ev} - \xi_1 = \frac{V_{Gap3_1} + V_{Gap3_2}}{RCn(n + \alpha)\zeta} \quad (6.66)$$

where

$$\begin{aligned}
 V_{Gap3_1} = &3n^5 + (-16 + 24\alpha)n^4 + (-67 - 140\alpha + 56\alpha^2)n^3 + (-56 - 300\alpha - 408\alpha^2)n^2 \\
 &+ (-20 - 100\alpha - 224\alpha^2 - 240\alpha^3)n + (-24\alpha - 48\alpha^2)
 \end{aligned}$$

$$V_{Gap3_2} = (2n + 2\alpha) \cdot \psi^{1/2}$$

Obviously, if $V_{Gap3_1} \geq 0$, it can be found that $Gap3 > 0$. Else if $V_{Gap3_1} < 0$, $Gap3 > 0$

means that the following equation is correct:

$$V_{Gap3_1} + V_{Gap3_2} = \sqrt{V_{Gap3_2}^2} - \sqrt{V_{Gap3_1}^2} > 0. \quad (6.67)$$

It is obtained that

$$Gap4 = V_{Gap3_2}^2 - V_{Gap3_1}^2 \quad (6.68)$$

$$= n(n-1) \left[3n^4 + (12 + 24\alpha)n^3 + (17 + 56\alpha + 56\alpha^2)n^2 + (12 + 24\alpha)n + (16\alpha + 16\alpha^2) \right] \cdot V_{Gap3_3}$$

with

$$\begin{aligned}
 V_{Gap3_3} = &-3n^4 + (41 - 24\alpha)n^3 + (72 + 312\alpha - 56\alpha^2)n^2 \\
 &+ (-44 + 216\alpha + 760\alpha^2)n + (-48\alpha + 144\alpha^2 + 480\alpha^3)
 \end{aligned} \quad (6.69)$$

When $V_{Gap3_1} < 0$, there is $V_{Gap3_3} > -V_{Gap3_1} > 0$. Therefore, if $Gap5$ defined in eqn. (6.70)

is more than 0, it can be concluded that $V_{Gap3_3} > 0$.

$$Gap5 = nV_{Gap3_3} - (-V_{Gap3_1}) \quad (6.70)$$

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

The “GapN” defined above can be described as a polynomial of n :

$$Gap = f_m(\alpha)n^m + f_{m-1}(\alpha)n^{m-1} + f_{m-2}(\alpha)n^{m-2} + \cdots + f_1(\alpha)n + f_0(\alpha) \quad (6.71)$$

where $f_k(\alpha)$ is the polynomial of α , $k = 0, 1, 2, \dots, m$ and $f_m(\alpha) > 0$. It will be verified in the following that $Gap > 0$.

Since $f_m(\alpha) > 0$, let $k = m$, $Gap_k = Gap$ and $f'_m(\alpha) = f_m(\alpha)$. If $f'_k(\alpha) \geq 0$, set $f'_{k-1}(\alpha) = 2f'_k(\alpha) + f_{k-1}(\alpha)$. Thus when $k = m$, it can be obtained that

$$Gap_{m-1} = f'_m(\alpha)n^{m-1} + \sum_{i=0}^{m-2} f_i(\alpha)n^i$$

and $Gap_m > Gap_{m-1}$. If $Gap_0 \geq 0$, because $Gap = Gap_m > Gap_{m-1} \geq \cdots \geq Gap_1 \geq Gap_0$, it can be obtained that $Gap > 0$. Considering this mathematical phenomena, eqn. (6.70) can be expressed by

$$\begin{aligned} Gap5 &= nV_{Gap3_3} - (-V_{Gap3_1}) \quad (6.72) \\ &= 25n^4 + (5 + 172\alpha)n^3 + (-100 - 84\alpha + 352\alpha^2)n^2 \\ &\quad + (-20 - 148\alpha - 80\alpha^2 + 240\alpha^3)n + (-24\alpha - 48\alpha^2) \\ &> (25 \cdot 2 + 5 + 172\alpha)n^3 + (-100 - 84\alpha + 352\alpha^2)n^2 \\ &\quad + (-20 - 148\alpha - 80\alpha^2 + 240\alpha^3)n + (-24\alpha - 48\alpha^2) \\ &> [2(25 \cdot 2 + 5 + 172\alpha) + (-100 - 84\alpha + 352\alpha^2)]n^2 \\ &\quad + (-20 - 148\alpha - 80\alpha^2 + 240\alpha^3)n + (-24\alpha - 48\alpha^2) \\ &= (10 + 260\alpha + 352\alpha^2)n^2 + (-20 - 148\alpha - 80\alpha^2 + 240\alpha^3)n + (-24\alpha - 48\alpha^2) \\ &> [2(10 + 260\alpha + 352\alpha^2) + (-20 - 148\alpha - 80\alpha^2 + 240\alpha^3)]n + (-24\alpha - 48\alpha^2) \\ &= (372\alpha + 624\alpha^2 + 240\alpha^3)n + (-24\alpha - 48\alpha^2) \\ &\geq 2(372\alpha + 624\alpha^2 + 240\alpha^3) + (-24\alpha - 48\alpha^2) \geq 720\alpha + 1200\alpha^2 + 480\alpha^3 \geq 0 \end{aligned}$$

From eqn. (6.72), we can conclude that, when $V_{Gap3_1} < 0$, there is $V_{Gap3_3} > -V_{Gap3_1} > 0$.

Then eqn. (6.68) gives that $V_{Gap3_2}^2 - V_{Gap3_1}^2 > 0$ and eqn. (6.67) is correct. Thus, considering eqn. (6.66), eqn. (6.63) can also be proven correct. ■

Now that eqns. (6.62) and (6.63) are verified, Theorem 6.1 is also proven.

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

(B) $x_k(\xi_1)$ Improvement Proof

From eqn. (6.56), the eigenvector $\mathbf{x}_\xi$ which corresponds to ξ_1 is attained:

$$\mathbf{x}_\xi = \begin{bmatrix} x_1(\xi_1) \\ x_2(\xi_1) \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{1+\varphi_{\xi_1}^2}} \\ \frac{1}{\sqrt{1+\varphi_{\xi_1}^{-2}}} \end{bmatrix} \quad \text{with} \quad \varphi_{\xi_1} = \frac{\xi_1 - d_{11}}{d_{12}} \quad (6.73)$$

where d_{11} as well as d_{12} are given in eqn. (6.43) and ξ_1 is given by eqns. (6.49).

Theorem 6.2: Corresponding to ξ_1 , the components of the eigenvector $\mathbf{x}_\xi$, $x_1(\xi_1)$ and $x_2(\xi_1)$, lie within the bounds of eigenvector components bounds by Marinov and Shivakumar (1996) and $x_1(\xi_1)$ is larger than $x_2(\xi_1)$, i.e.

$$\underline{r} < x_2(\xi_1) < x_1(\xi_1) < \bar{r} \quad (6.74)$$

where $x_1(\xi_1)$ and $x_2(\xi_1)$ are given by eqns. (6.73) and $\underline{r}$ as well as $\bar{r}$ are given by eqn. (6.59).

Theorem 6.2 can be expressed as

$$\text{Case 1: } x_1(\xi_1) > x_2(\xi_1) \quad (6.75)$$

$$\text{Case 2: } x_1(\xi_1) < \bar{r} \quad (6.76)$$

$$\text{Case 3: } x_2(\xi_1) > \underline{r} \quad (6.77)$$

Proof of Case 1 of Theorem 6.2:

To find the smaller one between $x_1(\xi_1)$ and $x_2(\xi_1)$, the value of $\varphi_{\xi_1}^2$ is studied. From eqns. (6.43), (6.45), (6.46), (6.49) and (6.73), it is derived that

$$\varphi_{\xi_1} = \frac{\varphi_{\xi_1-1}}{\varphi_{\xi_1-2}} \quad (6.78)$$

where

$$\begin{aligned} \varphi_{\xi_1-1} = & 2 \left[14n^3 + (42 + 84\alpha)n^2 + (34 + 120\alpha + 120\alpha^2)n + 24\alpha + 12 - \psi^{1/2} \right] \\ & + 4(n^2 - 4)(n + 6\alpha + 3) \end{aligned}$$

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

$$\varphi_{\xi_1-2} = 4\sqrt{15}(n+4\alpha+2)\tau^{1/2}$$

Then $Gap6$ is characterized and computed as

$$Gap6 = \varphi_{\xi_1-1}^2 - \varphi_{\xi_1-2}^2 = V_{Gap6-1} - V_{Gap6-2} \quad (6.79)$$

where

$$\begin{aligned} V_{Gap6-1} &= 728n^6 + (5808 + 8016\alpha)n^5 + (16808 + 52128\alpha + 30528\alpha^2)n^4 \\ &\quad + (20496 + 112752\alpha + 168480\alpha^2 + 49920\alpha^3)n^3 \\ &\quad + (7712 + 82656\alpha + 250656\alpha^2 + 249600\alpha^3 + 38400\alpha^4)n^2 \\ &\quad + (-2112 - 3264\alpha + 46080\alpha^2 + 145920\alpha^3 + 115200\alpha^4)n \\ &\quad + (-768 - 12672\alpha - 51072\alpha^2 - 76800\alpha^3 - 38400\alpha^4) \\ V_{Gap6-2} &= [128n^3 + (384 + 768\alpha)n^2 + (208 + 960\alpha + 960\alpha^2)n - (96 + 192\alpha)] \cdot \psi^{1/2} \end{aligned}$$

Following the steps as in eqn. (6.72), it is found that both V_{Gap6-1} and V_{Gap6-2} are larger than 0. Also, $Gap7$ shown in eqn. (6.80) can be proven larger than 0.

$$Gap7 = V_{Gap6-2}^2 - V_{Gap6-1}^2 > 0 \quad (6.80)$$

Hence, eqns. (6.79) and (6.80) give that

$$\varphi_{\xi_1-1}^2 < \varphi_{\xi_1-2}^2 \quad \text{and} \quad \varphi_{\xi_1}^2 = \frac{\varphi_{\xi_1-1}^2}{\varphi_{\xi_1-2}^2} < 1 \quad (6.81)$$

Thus, eqn. (6.75) can be proven. ■

Proof of Case 2 of Theorem 6.2:

From eqns. (6.59) and (6.73), it is obvious that $x_1(\xi_1) < \bar{r} = 1$. Hence, eqn. (6.76) is proven. ■

Proof of Case 3 of Theorem 6.2:

Since $x_1(\xi_1) > x_2(\xi_1)$, eqn. (6.77) can be expressed by

$$x_2(\xi_1) = \frac{1}{\sqrt{1 + \varphi_{\xi_1}^{-2}}} > \underline{r} = \frac{2}{n(n+1+2\alpha)} \quad (6.82)$$

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

Eqn. (6.82) can be transformed into

$$Gap8 = \varphi_{\xi_1-1}^2 [n(n+1+2\alpha)]^2 - 4(\varphi_{\xi_1-1}^2 + \varphi_{\xi_1-2}^2) > 0 \quad (6.83)$$

Following the steps as in eqn. (6.72), eqn. (6.83) can be proven correct. So, eqn. (6.82) is also proven. Thus, eqn. (6.77) can be proven. ■

Since eqns. (6.75)-(6.77) are verified correct, Theorem 6.2 is proven.

(C) ξ_2 Improvement Proof

Theorem 6.3:

If the network parameters satisfy the following criteria:

$$V_C = 3n^4 + (-41 + 24\alpha)n^3 + (-72 - 312\alpha + 56\alpha^2)n^2 + (44 - 216\alpha - 760\alpha^2)n + (48\alpha - 144\alpha^2 - 480\alpha^3) > 0 \quad (6.84)$$

the second eigenvalue of $\hat{\mathbf{A}}_i^T$, ξ_2 , is useful and lies within the eigenvalue bounds by Marinov and Shivakumar (1996), i.e.

$$\underline{q}_{ev} < \xi_2 < \bar{q}_{ev} \quad (6.85)$$

else, ξ_2 is regarded as useless eigenvalue and discarded. Here, ξ_2 is given by eqns. (6.50) and $\underline{q}_{ev}$ as well as $\bar{q}_{ev}$ are given by eqn. (6.58).

Theorem 6.3 can be expressed as

$$\text{Case 1: When } V_C > 0, \xi_2 - \underline{q}_{ev} > 0. \quad (6.86)$$

$$\text{Case 2: When } V_C > 0, \bar{q}_{ev} - \xi_2 > 0 \quad (6.87)$$

Case 3: When $V_C \leq 0$, ξ_2 is regarded as useless eigenvalue.

Proof of Case 1 of Theorem 6.3:

Eqns. (6.50) and (6.58) give

$$Gap9 = \xi_2 - \underline{q}_{ev} = \frac{2(V_{Gap9-1} + V_{Gap9-2})}{RCn(n+1+2\alpha)\varsigma} \quad (6.88)$$

where ς is defined in eqn. (6.46) and

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

$$V_{Gap9_1} = 11n^4 + (44 + 88\alpha)n^3 + (59 + 232\alpha + 232\alpha^2)n^2 \\ + (34 + 188\alpha + 360\alpha^2 + 240\alpha^3)n + (8 + 32\alpha + 32\alpha^2)$$

$$V_{Gap9_2} = (n + 1 + 2\alpha) \cdot \psi^{1/2}$$

Obviously, both V_{Gap9_1} and V_{Gap9_2} are larger than 0 in any case. Thus, eqn. (6.86) can be proven. ■

Proof of Case 2 of Theorem 6.3:

From eqns. (6.50) and (6.58), we have

$$Gap10 = \bar{q}_{ev} - \xi_2 = \frac{V_{Gap10_1} - V_{Gap10_2}}{RCn(n + \alpha)\zeta} \quad (6.89)$$

where

$$V_{Gap10_1} = 3n^5 + (-16 + 24\alpha)n^4 + (-67 - 140\alpha + 56\alpha^2)n^3 \quad (6.90)$$

$$- (56 + 300\alpha + 408\alpha^2)n^2 - (20 + 100\alpha + 224\alpha^2 + 240\alpha^3)n - (24\alpha + 48\alpha^2)$$

$$V_{Gap10_2} = (2n + 2\alpha)\psi^{1/2}$$

It is noticed that, when $V_{Gap10_1} \leq 0$, $Gap10$ shown in eqn. (6.89) must be less than 0. In this case, there is $\xi_2 > \bar{q}$ so that ξ_2 is out of the original bound of eigenvalue and can be regarded as useless value. When $V_{Gap10_1} > 0$, eqn. (6.89) can be expressed by

$$Gap10 = \bar{q}_{ev} - \xi_2 = \frac{\sqrt{V_{Gap10_1}^2} - \sqrt{V_{Gap10_2}^2}}{RCn(n + \alpha)\zeta} \quad (6.91)$$

$Gap11$ is defined and extended as

$$Gap11 = V_{Gap10_1}^2 - V_{Gap10_2}^2 \quad (6.92)$$

$$= V_C \cdot \left\{ n(n-1) \left[3n^4 + (12 + 24\alpha)n^3 + (17 + 56\alpha + 56\alpha^2)n^2 + (12 + 24\alpha)n + (16\alpha + 16\alpha^2) \right] \right\}$$

where V_C is same as in eqn. (6.84):

$$V_C = 3n^4 + (-41 + 24\alpha)n^3 + (-72 - 312\alpha + 56\alpha^2)n^2 \\ + (44 - 216\alpha - 760\alpha^2)n + (48\alpha - 144\alpha^2 - 480\alpha^3)$$

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

From eqns. (6.91) and (6.92), it is obtained that, only when $V_C > 0$, there exists $\xi_2 < \bar{q}_{ev}$.

Thus, eqn. (6.87) can be proven. ■

Proof of Case 3 of Theorem 6.3:

From eqns. (6.91) and (6.92), it is found that, when $V_C \leq 0$, ξ_2 is larger than $\bar{q}_{ev}$. So, ξ_2 is regarded as useless and should be discarded. ■

Since all three cases are verified correct, Theorem 6.3 is proven.

(D) $x_k(\xi_2)$ Improvement Proof

From eqn. (6.56), the eigenvector $\mathbf{x}_\xi$ which corresponds to ξ_2 is attained:

$$\mathbf{x}_\xi = \begin{bmatrix} x_1(\xi_2) \\ x_2(\xi_2) \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{1+\varphi_{\xi_2}^2}} \\ \frac{1}{\sqrt{1+\varphi_{\xi_2}^{-2}}} \end{bmatrix} \quad \text{with} \quad \varphi_{\xi_2} = \frac{\xi_2 - d_{11}}{d_{12}} \quad (6.93)$$

where d_{11} as well as d_{12} are given in eqn. (6.43) and ξ_2 is given by eqns. (6.50).

Theorem 6.4: Corresponding to ξ_2 , the components of the eigenvector $\mathbf{x}_\xi$, $x_1(\xi_2)$ and $x_2(\xi_2)$, lie within the bounds of eigenvector components bounds by Marinov and Shivakumar (1996) and $x_2(\xi_2)$ is larger than $x_1(\xi_2)$, i.e.

$$\underline{r} < x_1(\xi_2) < x_2(\xi_2) < \bar{r} \quad (6.94)$$

where $x_1(\xi_2)$ and $x_2(\xi_2)$ are given by eqns. (6.93) and $\underline{r}$ as well as $\bar{r}$ are given by eqn. (6.59).

Theorem 6.4 can be expressed as

$$\text{Case 1: } x_2(\xi_2) > x_1(\xi_2) \quad (6.95)$$

$$\text{Case 2: } x_2(\xi_2) < \bar{r} \quad (6.96)$$

$$\text{Case 3: } x_1(\xi_2) > \underline{r} \quad (6.97)$$

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

Proof of Case 1 of Theorem 6.4:

To compare $x_1(\xi_2)$ and $x_2(\xi_2)$, the value of $\varphi_{\xi_2}^2$ is studied. Similar to the steps to prove Case 1 of Theorem 6.2, Eqns. (6.43), (6.45), (6.46), (6.49) and (6.93) give

$$\varphi_{\xi_2} = \frac{\varphi_{\xi_2-1}}{\varphi_{\xi_2-2}} \quad (6.98)$$

where

$$\begin{aligned} \varphi_{\xi_2-1} &= 2 \left[14n^3 + (42 + 84\alpha)n^2 + (34 + 120\alpha + 120\alpha^2)n + 24\alpha + 12 + \psi^{1/2} \right] \\ &\quad + 4(n^2 - 4)(n + 6\alpha + 3) \\ \varphi_{\xi_2-2} &= 4\sqrt{15}(n + 4\alpha + 2)\tau^{1/2} \end{aligned}$$

Then define $Gap12$ as

$$Gap12 = \varphi_{\xi_2-1}^2 - \varphi_{\xi_2-2}^2 = V_{Gap6-1} + V_{Gap6-2} \quad (6.99)$$

where V_{Gap6-1} and V_{Gap6-2} are same in eqn. (6.79). Since both V_{Gap6-1} and V_{Gap6-2} have been proven larger than 0, it is obtained:

$$\varphi_{\xi_2}^2 = \frac{\varphi_{\xi_2-1}^2}{\varphi_{\xi_2-2}^2} > 1 \quad (6.100)$$

As a result, it is found that $x_2(\xi_2) > x_1(\xi_2)$. Thus, Case 1 is proven. ■

Proof of Case 2 of Theorem 6.4:

From eqns. (6.59) and (6.93), it is obvious that $x_2(\xi_2) < \bar{r} = 1$. Thus, eqn. (6.96) is proven. ■

Proof of Case 3 of Theorem 6.4:

In view of eqns. (6.59) and (6.97), the item waiting for verification becomes

$$x_{\xi_2-1} > \underline{r} = 1/\beta = \frac{2}{n(n+1+2\alpha)} \quad (6.101)$$

Eqn. (6.101) is equivalent to

$$Gap13 = \varphi_{\xi_2-2}^2 \cdot [n(n+1+2\alpha)]^2 - 4(\varphi_{\xi_2-1}^2 + \varphi_{\xi_2-2}^2) > 0 \quad (6.102)$$

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

Substituting the value of φ_{ξ_2-1} and φ_{ξ_2-2} in eqn. (6.98) into eqn. (6.102), eqn. (6.102) can be proven proper. As a consequence, eqn. (9.101) is also verified to be correct. Thus, eqn. (6.97) can be proven. ■

Since eqns. (6.95)-(6.97) are verified correct, Theorem 6.4 is proven.

6.2.6 Optimized Global Switching Time Bounds of RC-Ladder Networks via Lanczos Process

In the above section, using the new method via Lanczos process, the eigenvalues and corresponding eigenvector components of $\hat{\mathbf{A}}_i^T$, ξ and $x_k(\xi)$, can be obtained. Then it has been proved that they not only satisfy eqns. (6.58) and (6.59) respectively, but also have tighter range; that is clearly represented in Theorems 6.1-6.4 (Section 6.2.5).

In eqn. (6.11), the new bounds of the eigenvalues and eigenvectors of $\mathbf{A}_i^T$, are denoted as $\underline{q}_{ev_LP}$, $\bar{q}_{ev_LP}$, $\underline{r}_{LP}$ and $\bar{r}_{LP}$, can be deduced as follows, using eqns. (6.61), (6.74), (6.84), (6.85) and (6.94):

(i) When the criteria shown in eqn. (6.84) is satisfied,

$$V_C > 0, \quad (6.84)$$

then

$$\underline{q}_{ev_LP} = \xi_1, \quad \bar{q}_{ev_LP} = \xi_2, \quad (6.103)$$

$$\underline{r}_{LP} = \min \{x_2(\xi_1), x_1(\xi_2)\} \quad \text{and} \quad \bar{r}_{LP} = \max \{x_1(\xi_1), x_2(\xi_2)\} \quad (6.104)$$

Substituting eqns. (6.73) and (6.93), eqn. (6.104) is transformed into

$$\underline{r}_{LP} = \min \left\{ \frac{1}{\sqrt{1+\varphi_{\xi_1}^{-2}}}, \frac{1}{\sqrt{1+\varphi_{\xi_2}^2}} \right\} \quad \text{and} \quad \bar{r}_{LP} = \max \left\{ \frac{1}{\sqrt{1+\varphi_{\xi_1}^2}}, \frac{1}{\sqrt{1+\varphi_{\xi_2}^{-2}}} \right\} \quad (6.105)$$

where φ_{ξ_1} and φ_{ξ_2} are defined in eqn. (6.73) and (6.93).

(ii) In other conditions, ξ_2 is incorrect and abandoned; therefore we have

$$\underline{q}_{ev_LP} = \bar{q}_{ev_LP} = \xi_1, \quad \underline{r}_{LP} = x_2(\xi_1) \quad \text{and} \quad \bar{r}_{LP} = x_1(\xi_1) \quad (6.106)$$

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

Substituting eqns. (6.73) into eqn. (6.106), it is derived that

$$\underline{q}_{ev_LP} = \bar{q}_{ev_LP} = \xi_1, \quad \underline{r}_{LP} = \frac{1}{\sqrt{1+\varphi_{\xi_1}^{-2}}} \quad \text{and} \quad \bar{r}_{LP} = \frac{1}{\sqrt{1+\varphi_{\xi_1}^2}}. \quad (6.107)$$

After obtaining $\underline{q}_{ev_LP}$, $\bar{q}_{ev_LP}$, $\underline{r}_{LP}$ and $\bar{r}_{LP}$, the bounds of GST generated by the new method using eqn. (6.57) are shown to be

$$\underline{T}_{\lambda_LP} = \max \left\{ 0; -\frac{1}{\bar{q}_{ev_LP}} \ln \frac{\lambda \bar{r}_{LP}}{\underline{r}_{LP}} \right\} \quad \text{and} \quad \bar{T}_{\lambda_LP} = -\frac{1}{\underline{q}_{ev_LP}} \ln \frac{\lambda \underline{r}_{LP}}{\bar{r}_{LP}} \quad (6.108)$$

Because it has been proved that $\underline{q}_{ev} < \underline{q}_{ev_LP} < \bar{q}_{ev_LP} < \bar{q}_{ev}$ and $\underline{r} < \underline{r}_{LP} < \bar{r}_{LP} < \bar{r}$, the new GST bounds must be within the original GST bounds and much tighter than those. Thus, from eqns. (6.103)-(6.107), the optimized bounds of GST can be expressed as follows:

(a) When the criteria shown in eqn. (6.84) is satisfied,

$$\begin{aligned} V_C = 3n^4 + (-41 + 24\alpha)n^3 + (-72 - 312\alpha + 56\alpha^2)n^2 \\ + (44 - 216\alpha - 760\alpha^2)n + (48\alpha - 144\alpha^2 - 480\alpha^3) > 0 \end{aligned} \quad (6.109)$$

Thus,

$$\underline{T}_{\lambda_LP} = \max \left\{ 0; -\frac{1}{\xi_2} \ln \frac{\lambda \max \left\{ \sqrt{1+\varphi_{\xi_1}^{-2}}, \sqrt{1+\varphi_{\xi_2}^2} \right\}}{\min \left\{ \sqrt{1+\varphi_{\xi_1}^2}, \sqrt{1+\varphi_{\xi_2}^{-2}} \right\}} \right\} \quad (6.110)$$

$$\bar{T}_{\lambda_LP} = -\frac{1}{\xi_1} \ln \frac{\lambda \min \left\{ \sqrt{1+\varphi_{\xi_1}^2}, \sqrt{1+\varphi_{\xi_2}^{-2}} \right\}}{\max \left\{ \sqrt{1+\varphi_{\xi_1}^{-2}}, \sqrt{1+\varphi_{\xi_2}^2} \right\}} \quad (6.111)$$

(b) In other conditions, there are

$$\underline{T}_{\lambda_LP} = \max \left\{ 0; -\frac{1}{\xi_1} \ln \frac{\lambda \sqrt{1+\varphi_{\xi_1}^{-2}}}{\sqrt{1+\varphi_{\xi_1}^2}} \right\} \quad (6.112)$$

$$\bar{T}_{\lambda_LP} = -\frac{1}{\xi_1} \ln \frac{\lambda \sqrt{1+\varphi_{\xi_1}^2}}{\sqrt{1+\varphi_{\xi_1}^{-2}}} \quad (6.113)$$

where ξ_1 and ξ_2 are respectively represented in eqns. (6.49) and (6.50), and φ_{ξ_1} and φ_{ξ_2} presented in eqns. (6.73) and (6.93) can be expanded into the following forms,

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

$$\varphi_{\xi_1} = \frac{[16n^3 + (48 + 96\alpha)n^2 + (26 + 120\alpha + 120\alpha^2)n + (-12 - 24\alpha)] - \psi^{1/2}}{2\sqrt{15}(n + 4\alpha + 2)\tau^{1/2}} \quad (6.114)$$

$$\varphi_{\xi_2} = \frac{[16n^3 + (48 + 96\alpha)n^2 + (26 + 120\alpha + 120\alpha^2)n + (-12 - 24\alpha)] + \psi^{1/2}}{2\sqrt{15}(n + 4\alpha + 2)\tau^{1/2}} \quad (6.115)$$

Here, τ and ψ are defined in eqns. (6.45) and (6.51), respectively.

6.3 Delay Time Bounds of RC-Ladder Networks via Lanczos Process

The delay of a network, T_D , can be considered as a special case of the GST, T_λ , i.e.

$$T_D = T_\lambda \quad \text{when } \lambda = \lambda_D \quad (6.116)$$

where the method to compute λ_D has been presented in Section 5.4. After obtaining the GST bounds and λ_D , the network delay time bounds can be easily computed by using λ_D to replace λ in the formulas of GST bounds.

Thus, with the GST bounds shown in (5.42) (in Section 5.3.3), which are based on the method given by Marinov and Shivakumar (1996), the delay time bounds can be computed as

$$\underline{T}_D = \max\{0; -RC(n + \alpha) \ln \lambda_D \beta\} \quad \text{and} \quad \bar{T}_D = -RC\beta \ln(\lambda_D / \beta) \quad (6.117)$$

where $\beta = \frac{n(n-1)}{2} + n(1 + \alpha)$.

For the new method via Lanczos process, the new delay bounds can be concluded from eqn. (6.110)-(6.113) as

If $V_C > 0$ as eqn. (6.109),

$$\underline{T}_{D_LP} = \max \left\{ 0; -\frac{1}{\xi_2} \ln \frac{\lambda_D \max \left\{ \sqrt{1 + \varphi_{\xi_1}^{-2}}, \sqrt{1 + \varphi_{\xi_2}^{-2}} \right\}}{\min \left\{ \sqrt{1 + \varphi_{\xi_1}^2}, \sqrt{1 + \varphi_{\xi_2}^2} \right\}} \right\} \quad (6.118)$$

$$\bar{T}_{D_LP} = -\frac{1}{\xi_1} \ln \frac{\lambda_D \min \left\{ \sqrt{1 + \varphi_{\xi_1}^2}, \sqrt{1 + \varphi_{\xi_2}^2} \right\}}{\max \left\{ \sqrt{1 + \varphi_{\xi_1}^{-2}}, \sqrt{1 + \varphi_{\xi_2}^{-2}} \right\}} \quad (6.119)$$

else,

$$\underline{T}_{D_LP} = \max \left\{ 0; -\frac{1}{\xi_1} \ln \frac{\lambda_D \sqrt{1 + \varphi_{\xi_1}^{-2}}}{\sqrt{1 + \varphi_{\xi_1}^2}} \right\} \quad (6.120)$$

$$\overline{T}_{D_LP} = -\frac{1}{\xi_1} \ln \frac{\lambda_D \sqrt{1 + \varphi_{\xi_1}^2}}{\sqrt{1 + \varphi_{\xi_1}^{-2}}} \quad (6.121)$$

where ξ_1 , ξ_2 , φ_{ξ_1} and φ_{ξ_2} are respectively given in eqns. (6.49), (6.50), (6.114) and (6.115).

6.4 Numerical Examples and Analysis

Numerical examples are given here to support the results obtained. Assuming that $RC = 1$, RC-ladder networks with different cell number n are selected as examples.

The method via Lanczos process is compared with not only the method by Marinov and Shivakumar (1996) but also with another improved method (Hou *et al.* 2000) in Section 6.4.1. Moreover, in their work (Marinov and Shivakumar 1996, Hou *et al.* 2000), the precision of numerical results are set at 10^{-1} and 10^{-4} , respectively. So, the numerical precision is set at 10^{-4} in the examples.

Some numerical examples will also be used to explain the criteria to select the approximation order of T_q when $q = 2$.

6.4.1 Another Method to Compute Improved Delay Time Bounds

For the same n -cell RC-ladder network, Hou *et al.* (2000) also gave an improved GST bound estimation method. The lower and upper bounds of q_{ev} , that are denoted as $\underline{q}_{ev_Hou}$ and $\overline{q}_{ev_Hou}$, are expressed as

$$\underline{q}_{ev_Hou} = \frac{1}{RCb_{ev2}} \quad \text{and} \quad \overline{q}_{ev_Hou} = \frac{1}{RCb_{ev1}} \quad (6.122)$$

where

$$b_{ev1} = \max \left\{ n + \alpha, \beta - \frac{1}{2 \left(1 + \cos \frac{(n-1)\pi}{n} \right)} - \frac{n}{2\pi} \left(tg \frac{(n-1)\pi}{2n} - tg \frac{\pi}{2n} \right) \right\} \quad (6.123)$$

$$b_{ev2} = \beta - \frac{n+1}{2\pi} tg \frac{(n-1)\pi}{2(n+1)} \quad \text{and} \quad \beta = \frac{n(n-1)}{2} + n(1 + \alpha) \quad (6.124)$$

Also, the GST bounds are given by

$$\underline{T}_{\lambda_Hou} = \max \left\{ -RC \ln \lambda, -RC b_{ev1} \ln [\lambda (b_{ev2} - n + 1)] \right\} \quad (6.125)$$

$$\bar{T}_{\lambda_Hou} = -RC b_{ev2} \ln \frac{\lambda}{b_{ev2} - n + 1} \quad (6.126)$$

The corresponding delay bounds are obtained as

$$\underline{T}_{D_Hou} = \max \left\{ -RC \ln \lambda_D, -RC b_{ev1} \ln [\lambda_D (b_{ev2} - n + 1)] \right\} \quad (6.127)$$

$$\bar{T}_{D_Hou} = -RC b_{ev2} \ln \frac{\lambda_D}{b_{ev2} - n + 1} \quad (6.128)$$

6.4.2 Comparison of GST Bounds

The bounds of q_{ev} using the method by Marinov and Shivakumar (1996) are given in eqn. (6.58); those using the improved method by Hou *et al.* are given in eqn. (6.122), and those using the optimized method via Lanczos process are given in eqns. (6.103) and (6.106). As illustrated in Table 6.1 for $RC = 1$ and $\alpha = 2n$, a comparison of the bounds of q_{ev} by the three GST bound methods shows that the method via Lanczos process provides a tighter bound than the other two.

The GST bounds from the three methods are also compared. For the examples with different cell number n , each case has three subcases corresponding to three different α values (different loads): $\alpha = 0$, $\alpha = n$ and $\alpha = 2n$. The results are listed in Tables 6.2 - 6.7, and the bound width ε_{λ} is defined by

$$\varepsilon_{\lambda} = (\bar{T}_{\lambda} - \underline{T}_{\lambda}) / T_{\lambda} \quad (6.129)$$

Table 6.1 Comparison of eigenvalue bounds for RC-ladder network with $\alpha = 2n$

n		3	5	9	11
$\underline{q}_{ev}$	Method via Lanczos Process	0.0440	0.0163	0.0051	0.0035
	Method by Hou <i>et al.</i> (2000)	0.0428	0.0158	0.0049	0.0033
	Method by Marinov <i>et al.</i> (1996)	0.0417	0.0154	0.0048	0.0032
$\bar{q}_{ev}$	Method via Lanczos Process	0.0440	0.0163	0.0051	0.0035
	Method by Hou <i>et al.</i> (2000)	0.0445	0.0166	0.0052	0.0035
	Method by Marinov <i>et al.</i> (1996)	0.1111	0.0667	0.0370	0.0303
q_{ev}		0.0440	0.0163	0.0051	0.0035

Table 6.2 Comparison of GST bounds for RC-ladder network with $n = 3$ and $\lambda = 0.1$

$n = 3$ $\lambda = 0.1$	T_λ	GST Bound Estimation Method	$\underline{T}_\lambda$	$\bar{T}_\lambda$	ε_λ
$\alpha = 0$	11.2	Via Lanczos Process	8.3396	14.9031	0.5860
		Hou <i>et al.</i> (2000)	4.8474	18.8551	1.2507
		Marinov <i>et al.</i> (1996)	1.5	24.5	2.1
$\alpha = 3$	23.6	Via Lanczos Process	10.4380	53.0567	1.8059
		Hou <i>et al.</i> (2000)	2.3026	69.1931	2.8343
		Marinov <i>et al.</i> (1996)	0	75.1	3.2
$\alpha = 6$	35.9	Via Lanczos Process	6.3218	98.3906	2.5646
		Hou <i>et al.</i> (2000)	2.3026	125.3273	3.4269
		Marinov <i>et al.</i> (1996)	0	131.5	3.7

Table 6.3 Comparison of GST bounds for RC-ladder network with $n = 3$ and $\lambda = 0.01$

$n = 3$ $\lambda = 0.01$	T_λ	GST Bound Estimation Method	$\underline{T}_\lambda$	$\bar{T}_\lambda$	ε_λ
$\alpha = 0$	22.8	Via Lanczos Process	19.9610	26.5244	0.2879
		Hou <i>et al.</i> (2000)	15.0909	31.2048	0.7067
		Marinov <i>et al.</i> (1996)	8.4	38.3	1.3
$\alpha = 3$	55.3	Via Lanczos Process	42.1854	84.8041	0.7707
		Hou <i>et al.</i> (2000)	28.1135	102.2660	1.3409
		Marinov <i>et al.</i> (1996)	11.4	109.7	1.8
$\alpha = 6$	88.3	Via Lanczos Process	58.6779	150.7468	1.0427
		Hou <i>et al.</i> (2000)	34.6493	179.1235	1.6362
		Marinov <i>et al.</i> (1996)	12.8	186.8	2.0

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

Table 6.4 Comparison of GST bounds for RC-ladder network with $n = 5$ and $\lambda = 0.1$

$n = 5$ $\lambda = 0.1$	T_λ	GST Bound Estimation Method	$\underline{T}_\lambda$	$\bar{T}_\lambda$	ε_λ
$\alpha = 0$	26.8	Via Lanczos Process	19.6816	37.1252	0.6509
		Hou <i>et al.</i> (2000)	2.3026	60.5580	2.1737
		Marinov <i>et al.</i> (1996)	0	75.2	2.8
$\alpha = 5$	49.9	Via Lanczos Process	26.1962	141.3756	2.3082
		Hou <i>et al.</i> (2000)	2.3026	223.9051	4.4409
		Marinov <i>et al.</i> (1996)	0	239.6	4.8
$\alpha = 10$	73.2	Via Lanczos Process	14.4527	267.4947	3.4569
		Hou <i>et al.</i> (2000)	2.3026	404.5257	5.4949
		Marinov <i>et al.</i> (1996)	0	421.0	5.8

Table 6.5 Comparison of GST bounds for RC-ladder network with $n = 5$ and $\lambda = 0.01$

$n = 5$ $\lambda = 0.01$	T_λ	GST Bound Estimation Method	$\underline{T}_\lambda$	$\bar{T}_\lambda$	ε_λ
$\alpha = 0$	55.3	Via Lanczos Process	48.0851	65.5287	0.3154
		Hou <i>et al.</i> (2000)	24.1558	91.2883	1.2140
		Marinov <i>et al.</i> (1996)	9.5	109.6	1.8
$\alpha = 5$	133.7	Via Lanczos Process	109.9821	225.1615	0.8615
		Hou <i>et al.</i> (2000)	37.6085	312.2001	2.0538
		Marinov <i>et al.</i> (1996)	9.2	331.7	2.4
$\alpha = 10$	214.2	Via Lanczos Process	155.4264	408.4684	1.1813
		Hou <i>et al.</i> (2000)	31.4070	550.3853	2.4229
		Marinov <i>et al.</i> (1996)	6.46	570.7	2.6

Table 6.6 Comparison of GST bounds for RC-ladder network with $n = 9$ and $\lambda = 0.1$

$n = 9$ $\lambda = 0.1$	T_λ	GST Bound Estimation Method	$\underline{T}_\lambda$	$\bar{T}_\lambda$	ε_λ
$\alpha = 0$	78.5	Via Lanczos Process	56.4442	112.2475	0.7109
		Hou <i>et al.</i> (2000)	2.3026	231.4468	2.9190
		Marinov <i>et al.</i> (1996)	0	274.9	3.5
$\alpha = 9$	108.4	Via Lanczos Process	78.0137	447.0303	3.4042
		Hou <i>et al.</i> (2000)	2.3026	851.4507	7.8335
		Marinov <i>et al.</i> (1996)	0	899.5	8.3
$\alpha = 18$	143.0	Via Lanczos Process	38.1577	857.2412	5.7279
		Hou <i>et al.</i> (2000)	2.3026	1530.1	10.684
		Marinov <i>et al.</i> (1996)	0	1580.5	11.1

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

Table 6.7 Comparison of GST bounds for RC-ladder network with $n = 9$ and $\lambda = 0.01$

$n = 9$ $\lambda = 0.01$	T_λ	GST Bound Estimation Method	$\underline{T}_\lambda$	$\bar{T}_\lambda$	ε_λ
$\alpha = 0$	162.9	Via Lanczos Process	140.7900	196.5934	0.3426
		Hou <i>et al.</i> (2000)	32.7677	323.7844	1.7865
		Marinov <i>et al.</i> (1996)	7.2	378.5	2.3
$\alpha = 9$	371.0	Via Lanczos Process	340.5356	709.5523	0.9947
		Hou <i>et al.</i> (2000)	4.6052	1130.3	3.0342
		Marinov <i>et al.</i> (1996)	0	1189.6	3.2
$\alpha = 18$	590.7	Via Lanczos Process	485.8572	1304.9406	1.3866
		Hou <i>et al.</i> (2000)	4.6052	1995.5	3.3703
		Marinov <i>et al.</i> (1996)	0	2057.1	3.5

From Tables 6.2 - 6.7, it is obvious that the GST bounds from the optimized method via Lanczos process are much tighter than those from the other two methods. This fact is also illustrated by the comparison of the bound width ε_λ . Moreover, it is evident from the above examples that no matter which method is used, the obtained GST bounds are tighter for smaller loads (smaller α), for fewer cells and for a small λ .

6.4.3 Comparison of Delay Bounds

As mentioned in Section 5.4, by letting $\lambda = \lambda_D$, the network delay time, T_D , can be computed. The delay time bounds generated by the method via Lanczos process are then compared with those produced by the other two methods. Each n value has five corresponding subcases with different α values (different loads). The results are listed in Tables 6.8 and 6.9, the bound width ε_D is defined by

$$\varepsilon_D = (\bar{T}_D - \underline{T}_D) / T_D \quad (6.124)$$

Different from Tables 6.2 to 6.7, Tables 6.8 and 6.9 also include some examples having $V_C > 0$. Comparing the delay time bounds generated by the three methods in Tables 6.8 and 6.9, the improvement of the method via Lanczos process is obvious, whether $V_C > 0$ or not.

It can be observed from Tables 6.5 and 6.6 that the delay bounds from the method via

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

Lanczos process are much tighter than those from the other methods. Comparing the bound width ε_D , the same conclusion can also be derived.

Table 6.8 Comparison of delay bounds for RC-ladder network with $n = 20$

$n = 20$	V_C	λ_D	T_D (s)	Bound Esti Method	$\underline{T}_D$ (s)	$\overline{T}_D$ (s)	ε_D
$\alpha = 0$	>0	0.3249	159.0371	Via Lanczos	8.1744	329.1470	2.0182
				Hou <i>et al.</i>	1.1242	1174.4789	7.3779
				Marinov <i>et al.</i>	0	1358.9668	8.5450
$\alpha = 10$	>0	0.2889	302.7172	Via Lanczos	0	893.9365	2.9530
				Hou <i>et al.</i>	1.2415	2773.6749	9.1585
				Marinov <i>et al.</i>	0	29756.5486	9.8298
$\alpha = 20$	>0	0.2788	442.3352	Via Lanczos	0	1609.9305	3.6396
				Hou <i>et al.</i>	1.2773	4479.7499	10.1246
				Marinov <i>et al.</i>	0	4691.3684	10.6059
$\alpha = 30$	<0	0.2742	581.3079	Via Lanczos	0	2407.1649	4.1409
				Hou <i>et al.</i>	1.2938	6254.2688	10.7567
				Marinov <i>et al.</i>	0	6472.5614	11.1345
$\alpha = 40$	<0	0.2717	720.0760	Via Lanczos	0	3261.1449	4.5289
				Hou <i>et al.</i>	1.3032	8079.6912	11.2188
				Marinov <i>et al.</i>	0	8303.0988	11.5309

Table 6.9 Comparison of delay bounds for RC-ladder network with $n = 30$

$n = 30$	V_C	λ_D	T_D (s)	Bound Esti Method	$\underline{T}_D$ (s)	$\overline{T}_D$ (s)	ε_D
$\alpha = 0$	>0	0.3237	352.1986	Via Lanczos	17.4708	735.2703	2.0381
				Hou <i>et al.</i>	1.1279	2951.8279	8.3779
				Marinov <i>et al.</i>	0	3380.5021	9.5983
$\alpha = 30$	>0	0.2748	990.0455	Via Lanczos	0	3628.8624	3.6653
				Hou <i>et al.</i>	1.2919	11127.3200	11.2379
				Marinov <i>et al.</i>	0	11617.2160	11.7340
$\alpha = 60$	>0	0.2675	1615.0519	Via Lanczos	0	7364.2994	4.5598
				Hou <i>et al.</i>	1.3185	19968.2553	12.3630
				Marinov <i>et al.</i>	0	20484.2294	12.6833
$\alpha = 75$	<0	0.2659	1927.1720	Via Lanczos	0	9395.3373	4.8752
				Hou <i>et al.</i>	1.3247	24537.7579	12.7318
				Marinov <i>et al.</i>	0	25062.8816	13.0050
$\alpha = 90$	<0	0.2647	2239.2321	Via Lanczos	0	11507.2537	5.1389
				Hou <i>et al.</i>	1.3291	29183.3188	13.0321
				Marinov <i>et al.</i>	0	29716.1340	13.2707

6.4.4 Approximation Order Selection in Lanczos Process

Finally, some numerical examples are given here to explain the reason why the approximation order of $\mathbf{T}_q$ with $q=2$ is selected. In Section 6.2.3, it has been mentioned that with the value of k increasing, the parameters α_k , β_k and ρ_k have decreasing contribution to the performance of the reduced-order approximation model $\mathbf{T}_q$.

Choosing the RC-ladder network with $n=5$ and $\alpha=5$ as example, $\mathbf{T}_q$ can be computed for $q=2, 3$ and 4 . Then the corresponding $\hat{\mathbf{A}}_t^T$ can be obtained from eqn. (6.41). The eigenvalues and eigenvectors of $\hat{\mathbf{A}}_t^T$ for different q values are calculated as follows:

For $q=2$:

$$\mathbf{D} = \begin{bmatrix} 0.0275 & 0 \\ 0 & 0.3806 \end{bmatrix}, \mathbf{V} = \begin{bmatrix} 0.9796 & 0.4257 \\ 0.2011 & 0.9048 \end{bmatrix}$$

For $q=3$:

$$\mathbf{D} = \begin{bmatrix} 0.0275 & 0 & 0 \\ 0 & 0.4445 & 0 \\ 0 & 0 & 1.1826 \end{bmatrix}, \mathbf{V} = \begin{bmatrix} 0.9796 & 0.4009 & 0.2196 \\ 0.2011 & 0.8607 & 0.4890 \\ 0.0036 & 0.3138 & 0.8442 \end{bmatrix}$$

For $q=4$:

$$\mathbf{D} = \begin{bmatrix} 0.0275 & 0 & 0 & 0 \\ 0 & 0.4433 & 0 & 0 \\ 0 & 0 & 1.4461 & 0 \\ 0 & 0 & 0 & 2.2432 \end{bmatrix}, \mathbf{V} = \begin{bmatrix} 0.9796 & 0.1015 & 0.1881 & 0.1182 \\ 0.2011 & 0.8617 & 0.4204 & 0.2660 \\ 0.0036 & 0.3087 & 0.7890 & 0.5630 \\ 0.00002 & 0.0320 & 0.4067 & 0.7735 \end{bmatrix}$$

where the diagonal element of $\mathbf{D}$ are generalized eigenvalues, and the columns of the full matrix $\mathbf{V}$ are the corresponding eigenvectors for a unit constant.

Knowing that q_{ev} and r_k are the eigenvalue and eigenvector component of $\mathbf{A}_t^T$ respectively, the bounds of q_{ev} and r_k are computed by using the method of Marinov and Shivakumar (1996) as

Chapter 6: Optimized Delay Bounds of RC-Ladder Networks via Lanczos Process

$$0.025 \leq q_{ev} \leq 0.1 \quad \text{and} \quad 0.025 \leq r_k \leq 1.$$

Apparently, the useful values of q_{ev} and r_k converge when the order of $\mathbf{D}$ and $\mathbf{V}$ is two. It is proved that larger order of $\mathbf{T}_q$ will not bring more noticeable improvement. Hence, it is suitable to choose $q = 2$.

6.5 Summary

A simple method by Marinov and Shivakumar (1996) to calculate the GST bounds and delay time bounds of a *RC*-ladder network is represented in Chapter 5. For the same *RC*-ladder network, Hou *et al.* (2000) have reported an improved GST bound estimation method. To obtain more accurate GST bounds and delay time bounds of *RC*-ladder networks in shorter time, a new algorithm using the Lanczos process is proposed in this chapter. The Lanczos process is used to obtain the reduced-order model of a lumped circuit with little numerical degradation. The performance of an *RC*-ladder network can be described with only a 2nd-order model.

Comparing the GST bounds and delay bounds from the three methods, it is easily seen that the GST and delay bounds from the optimized method via Lanczos process are much tighter than those from the other two methods. Moreover, it is evident that no matter which method is used, the obtained bounds are tighter for smaller loads and for fewer cells.

Chapter 7: Stopping Criteria of Delay Evaluation for RLC -Ladder Networks via Lanczos Process

7.1 Introduction

In Chapter 5, the delay time bound method (Marinov and Shivakumar 1996) for the RC -ladder network has been extended to the RLC -ladder network as shown in Figure 7.1, which is composed of n RLC cells, capacitively loaded by C_t , and excited at $t = 0$ by a constant voltage source. This network is same as that in Figure 5.3 and repeated here for convenience. Moreover, in Chapter 6, an optimized method via Lanczos process has been applied to compute improved delay time bounds of RC -ladder networks. In this chapter, the optimized method via Lanczos process will be extended to the RLC -ladder network in Figure 7.1 for computing the delay time bounds. These delay time bounds will be compared with those obtained from the method in Chapter 5.

Figure 7.1 An n -cell RLC -ladder network with capacitive load (same as Figure 5.3 and repeated for convenience)

7.2 Optimized Algorithm via Lanczos Process to Compute Delay Time Bounds of RLC -Ladder Networks

7.2.1 Reduced-Order Approximation via Lanczos Process

In Section 6.2.1, two important parameters in Lanczos algorithm, $\mathbf{A}_L$ and $\mathbf{r}_L$, are defined. In Section 6.2.2, the relationship between $\mathbf{A}_L^T$ and $\mathbf{A}_L$ is described, and the new

Chapter 7: Stopping Criteria of Delay Evaluation for RLC-Ladder Networks via Lanczos Process

lower and upper bounds of q_{ev} and r_k via Lanczos process (LP), denoted by $\underline{q}_{ev_LP}$, $\bar{q}_{ev_LP}$, $\underline{r}_{LP}$ and $\bar{r}_{LP}$ are given such that

$$\underline{q}_{ev_LP} \leq q_{ev} \leq \bar{q}_{ev_LP} \quad \text{and} \quad \underline{r}_{LP} \leq r_k \leq \bar{r}_{LP} \quad (7.1)$$

The method to compute the reduced-order approximation of $\mathbf{A}_t^T$ by Lanczos process is given in Section 6.2.3. It is denoted by $\mathbf{T}_q$ with the structure:

$$\mathbf{T}_q = \begin{bmatrix} \alpha_1 & \beta_2 & 0 & \cdots & 0 \\ \rho_2 & \alpha_2 & \beta_3 & \ddots & \vdots \\ 0 & \rho_3 & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \beta_q \\ 0 & \cdots & 0 & \rho_q & \alpha_q \end{bmatrix}_{q \times q} \quad (7.2)$$

where α_k , β_k and ρ_k are the Lanczos parameters computed by Algorithm 6.1.

Again, a 2^{nd} -order model of $\mathbf{T}_q$ ($q = 2$) is used to approximate the $n \times n$ square matrix $\mathbf{A}_L$.

7.2.2 Approximation of Lanczos Process

To compute $\mathbf{T}_q$, the parameters in the Lanczos Algorithm, $\mathbf{l}$, $\mathbf{A}_L$ and $\mathbf{r}_L$, need to be obtained first. From eqns. (6.2) and (6.9),

$$\mathbf{l} = \mathbf{c} \quad \text{and} \quad \mathbf{A}_L = \mathbf{A}^{-1} = -\mathbf{A}_t^{-1} \quad (7.3)$$

Also, from eqns. (6.2) and (6.6), it can be derived that

$$\mathbf{r}_L = -\mathbf{A}_L \mathbf{b} \quad (7.4)$$

Using eqn. (3.1) (Chapter 3) and the state-space method in Section 2.3.1, the state parameters of the n -cell RLC-ladder network shown in Figure 7.1 are computed as

$$\mathbf{b} = \frac{1}{L} \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}_{2n \times 1}, \quad \mathbf{c}^T = [0 \quad 0 \quad \cdots \quad 0 \quad 1]_{1 \times 2n} \quad \text{and} \quad d = 0 \quad (7.5)$$

and $\mathbf{A}$ is presented in eqns. (5.13)-(5.15) (Chapter 5).

Chapter 7: Stopping Criteria of Delay Evaluation for RLC-Ladder Networks via Lanczos Process

Because $\mathbf{A}_t^{-1}$ has been computed and shown in eqn. (5.56), $\mathbf{A}_L$ is computed from eqn. (7.3). Then from eqns. (7.4) and (7.5), $\mathbf{r}_L$ is given as a $2n \times 1$ vector, i.e.

$$\mathbf{r}_L = [0 \ 1 \ 0 \ 1 \ \dots \ 0 \ 1]^T \quad (7.6)$$

Thus, from eqns. (7.3), (7.5) and (7.6), it can be shown that

$$\rho_1 = \|\mathbf{r}_L\|_2 = \sqrt{n}, \quad \eta_1 = \|\mathbf{l}\|_2 = 1, \quad (7.7)$$

$$\mathbf{v}_1 = \frac{\mathbf{r}_L}{\rho_1} = \frac{\sqrt{n}}{n} \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \\ 1 \end{bmatrix}_{2n \times 1} \quad \text{and} \quad \mathbf{w}_1 = \mathbf{l}/\eta_1 = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}_{2n \times 1} \quad (7.8)$$

Subsequently, based on the Lanczos algorithm presented in Algorithm 6.1 (Section 6.2.3), the 2nd-order approximation model of $\mathbf{A}_L$ is created. As mentioned in Section 5.3.5, $\mathbf{A}_t^{-1}$ has two different expressions: when $u=1$ and $u \neq 1$. Since $\mathbf{A}_L$ is the kernel parameter of Lanczos process, two situations are studied respectively in the following.

(A): When $u=1$

From eqns. (5.56), (5.58) and (7.3), $\mathbf{A}_L$ is obtained as

$$\mathbf{A}_L = \quad (7.9)$$

$$\begin{bmatrix} 0 & C & 0 & C & 0 & C & 0 & C & \dots & 0 & C(1+\alpha) \\ -L & -RC & 0 & -RC & 0 & -RC & 0 & -RC & \dots & 0 & -RC(1+\alpha) \\ 0 & 0 & 0 & C & 0 & C & 0 & C & \dots & 0 & C(1+\alpha) \\ -L & -RC & -L & -2RC & 0 & -2RC & 0 & -2RC & \dots & 0 & -2RC(1+\alpha) \\ 0 & 0 & 0 & 0 & 0 & C & 0 & C & \dots & 0 & C(1+\alpha) \\ -L & -RC & -L & -2RC & -L & -3RC & 0 & -3RC & \dots & 0 & -3RC(1+\alpha) \\ 0 & 0 & 0 & 0 & 0 & 0 & \ddots & \ddots & \vdots & \vdots & \vdots \\ -L & -RC & -L & -2RC & -L & -3RC & \ddots & 0 & C & 0 & C(1+\alpha) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \dots & -L & -RC(n-1) & 0 & -RC(n-1)(1+\alpha) \\ 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 & 0 & 0 & C(1+\alpha) \\ -L & -RC & -L & -2RC & -L & -3RC & \dots & -L & -RC(n-1) & -L & -nRC(1+\alpha) \end{bmatrix}_{2n \times 2n}$$

Chapter 7: Stopping Criteria of Delay Evaluation for RLC-Ladder Networks via Lanczos Process

From the Lanczos algorithm (Algorithm 6.1), each step produces the results as shown below:

For $k = 1$:

$$\text{Step (1): } \delta_1 = \mathbf{w}_1^T \mathbf{v}_1 = \frac{\sqrt{n}}{n} \quad (7.10)$$

$$\text{Step (2): } \alpha_1 = \frac{\mathbf{w}_1^T \mathbf{A}_L \mathbf{v}_1}{\delta_1} = \frac{-RC \cdot n(n+2\alpha+1)}{2}, \quad (7.11)$$

$$\beta_1 = \frac{\delta_1}{\delta_0} \cdot \eta_1 = \frac{\sqrt{n}}{n} \quad \text{and} \quad \gamma_1 = \frac{\delta_1}{\delta_0} \cdot \rho_1 = 1 \quad (7.12)$$

Step (3): In the following, k_{odd} is the odd row number and k_{even} is the even row number.

$$\mathbf{v} = \mathbf{A}_L \mathbf{v}_1 - \mathbf{v}_1 \alpha_1 - \mathbf{v}_0 \beta_1 = \begin{bmatrix} v_1 & v_2 & \cdots & v_{k_{\text{odd}}} & v_{k_{\text{even}}} & \cdots & v_{2n-1} & v_{2n} \end{bmatrix}^T \quad (7.13)$$

where $v_{k_{\text{odd}}}$ and $v_{k_{\text{even}}}$ are the odd and even components of $\mathbf{v}$, and they are

$$v_{k_{\text{odd}}} = \frac{C\sqrt{n}}{n} \left(n + \alpha - \frac{k_{\text{odd}} - 1}{2} \right) \quad (k_{\text{odd}} = 1, 3, \dots, 2n-1) \quad (7.14)$$

$$v_{k_{\text{even}}} = \frac{CR\sqrt{n}}{2n} \left[n(n+2\alpha+1) - \frac{k_{\text{even}}}{2} \left(2n - \frac{k_{\text{even}}}{2} + 2\alpha + 1 \right) \right] \quad (7.15)$$

$$(k_{\text{even}} = 2, 4, \dots, 2n)$$

Especially, when substituting $k_{\text{even}} = 2n$ into eqn. (7.15), we obtain $v_{2n} = 0$.

$$\begin{aligned} \mathbf{w} &= \mathbf{A}_L^T \mathbf{w}_1 - \mathbf{w}_1 \alpha_1 - \mathbf{w}_0 \gamma_1 \\ &= \begin{bmatrix} w_1 & w_2 & \cdots & w_{k_{\text{odd}}} & w_{k_{\text{even}}} & \cdots & w_{2n-1} & w_{2n} \end{bmatrix}^T \end{aligned} \quad (7.16)$$

where $w_{k_{\text{odd}}}$ and $w_{k_{\text{even}}}$ are the odd and even components of $\mathbf{w}$, and they are

$$w_{k_{\text{odd}}} = -L \quad (k_{\text{odd}} = 1, 3, 5, \dots, 2n-1) \quad (7.17)$$

$$w_{k_{\text{even}}} = \begin{cases} -\frac{k_{\text{even}} RC}{2} & (k_{\text{even}} = 2, 4, 6, \dots, 2n-2) \\ \frac{n(n-1) RC}{2} & (k_{\text{even}} = 2n) \end{cases} \quad (7.18)$$

Chapter 7: Stopping Criteria of Delay Evaluation for RLC-Ladder Networks via Lanczos Process

Step (4): Setting $i = \frac{k_{\text{odd}} - 1}{2}$ and $j = \frac{k_{\text{even}}}{2}$,

$$\rho_2 = \|\mathbf{v}\|_2 \quad (7.19)$$

$$\begin{aligned} &= \frac{C\sqrt{n}}{n} \sqrt{\sum_{i=0}^{n-1} (n + \alpha - i)^2 + \frac{R^2}{4} \sum_{j=1}^n [n(n + 2\alpha + 1) - j(2n - j + 2\alpha + 1)]^2} \\ &= C \sqrt{\frac{1}{6} [2n^2 + (6\alpha + 3)n + 6\alpha^2 + 6\alpha + 1] + \frac{R^2(n-1)}{60} [3n^3 + (15\alpha + 3)n^2 + (20\alpha^2 + 5\alpha - 2)n - (10\alpha^2 + 10\alpha + 2)]} \end{aligned}$$

$$\eta_2 = \|\mathbf{w}\|_2, \mathbf{v}_2 = \frac{\mathbf{v}}{\rho_2} \quad \text{and} \quad \mathbf{w}_2 = \frac{\mathbf{w}}{\eta_2} \quad (7.20)$$

For $k = 2$:

$$\text{Step (5): } \delta_2 = \mathbf{w}_2^T \mathbf{v}_2 = \frac{1}{\rho_2 \eta_2} \cdot \mathbf{w}^T \mathbf{v} = \frac{M}{\rho_2 \eta_2} \quad (7.21)$$

with

$$M = \mathbf{w}^T \mathbf{v} \quad (7.22)$$

$$= \frac{-C\sqrt{n}}{24} [R^2 C n^3 + (4\alpha + 2) R^2 C n^2 + (12L - R^2 C) n + (4\alpha + 2)(6L - 1R^2 C)]$$

$$\text{Step (6): } \alpha_2 = \frac{\mathbf{w}_2^T \mathbf{A}_L \mathbf{v}_2}{\delta_2} = \frac{\frac{1}{\rho_2 \eta_2} \mathbf{w}^T \mathbf{A}_L \mathbf{v}}{\delta_2} = \frac{\frac{1}{\rho_2 \eta_2} \mathbf{w}^T \mathbf{A}_L \mathbf{v}}{\frac{1}{\rho_2 \eta_2} \cdot M} = \frac{Q}{M} \quad (7.23)$$

with

$$Q = \mathbf{w}^T \mathbf{A}_L \mathbf{v} \quad (7.24)$$

$$= \frac{-RC^2(n^2 - 1)\sqrt{n}}{720} \cdot \left[R^2 C n^3 + 3(2\alpha + 1) R^2 C n^2 + 4(15L - R^2 C) n + 12(2\alpha + 1)(10L - R^2 C) \right]$$

$$\beta_2 = \frac{\delta_2}{\delta_1} \cdot \eta_2 = \frac{M\sqrt{n}}{\rho_2} \quad (7.25)$$

Step (7): Stop.

Chapter 7: Stopping Criteria of Delay Evaluation for RLC-Ladder Networks via Lanczos Process

(B): When $u \neq 1$

From eqns. (5.56), (5.57) and (7.3), we have

$$\mathbf{A}_L = \begin{bmatrix} 0 & C & 0 & C & 0 & C & 0 & C & \cdots & 0 & C(1+\alpha) \\ -L & -RC & 0 & -RC & 0 & -RC & 0 & -RC & \cdots & 0 & -RC(1+\alpha) \\ 0 & 0 & 0 & C & 0 & C & 0 & C & \cdots & 0 & C(1+\alpha) \\ -L & -RC & -L & -\frac{u^2-1}{u-1} \cdot RC & 0 & -\frac{u^2-1}{u-1} \cdot RC & 0 & -\frac{u^2-1}{u-1} \cdot RC & \cdots & 0 & -\frac{u^2-1}{u-1} \cdot RC(1+\alpha) \\ 0 & 0 & 0 & 0 & 0 & C & 0 & C & \cdots & 0 & C(1+\alpha) \\ -L & -RC & -L & -\frac{u^2-1}{u-1} \cdot RC & -L & -\frac{u^3-1}{u-1} \cdot RC & 0 & -\frac{u^3-1}{u-1} \cdot RC & \cdots & 0 & -\frac{u^3-1}{u-1} \cdot RC(1+\alpha) \\ 0 & 0 & 0 & 0 & 0 & 0 & \ddots & \ddots & \vdots & \vdots & \vdots \\ -L & -RC & -L & -\frac{u^2-1}{u-1} \cdot RC & -L & -\frac{u^3-1}{u-1} \cdot RC & \ddots & 0 & C & 0 & C(1+\alpha) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & -L & -\frac{u^{n-1}-1}{u-1} \cdot RC & 0 & -\frac{u^{n-1}-1}{u-1} \cdot RC(1+\alpha) \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & C(1+\alpha) \\ -L & -RC & -L & -\frac{u^2-1}{u-1} \cdot RC & -L & -\frac{u^3-1}{u-1} \cdot RC & \cdots & -L & -\frac{u^{n-1}-1}{u-1} \cdot RC & -L & -\frac{u^n-1}{u-1} \cdot RC(1+\alpha) \end{bmatrix}_{2n \times 2n}$$

Next, Algorithm 6.1 generates the following results:

$$\text{Step (1): } \delta_1 = \mathbf{w}_1^T \mathbf{v}_1 = \frac{\sqrt{n}}{n} \quad (7.27)$$

$$\text{Step (2): } \alpha_1 = \frac{\mathbf{w}_1^T \cdot \mathbf{A}_L \cdot \mathbf{v}_1}{\delta_1} = -RC \left[\frac{u^{n+1} - un - u + n}{(u-1)^2} + \frac{u^n - 1}{u-1} \cdot \alpha \right], \quad (7.28)$$

$$\beta_1 = \frac{\delta_1}{\delta_0} \cdot \eta_1 = \frac{\sqrt{n}}{n} \quad \text{and} \quad \gamma_1 = \frac{\delta_1}{\delta_0} \cdot \rho_1 = 1 \quad (7.29)$$

Step (3): Here, k_{odd} is the odd row number and k_{even} is the even row number.

$$\begin{aligned} \mathbf{v} &= \mathbf{A}_L \mathbf{v}_1 - \mathbf{v}_1 \alpha_1 - \mathbf{v}_0 \beta_1 \\ &= \begin{bmatrix} v_1 & v_2 & \cdots & v_{k_{\text{odd}}} & v_{k_{\text{even}}} & \cdots & v_{2n-1} & v_{2n} \end{bmatrix}^T \end{aligned} \quad (7.30)$$

where $v_{k_{\text{odd}}}$ and $v_{k_{\text{even}}}$ are the odd and even components of $\mathbf{v}$, and they are

$$v_{k_{\text{odd}}} = \frac{C\sqrt{n}}{n} \left(n + \alpha - \frac{k_{\text{odd}} - 1}{2} \right) \quad (k_{\text{odd}} = 1, 3, 5, \dots, 2n-1) \quad (7.31)$$

$$\begin{aligned} v_{k_{\text{even}}} &= \frac{RC\sqrt{n}}{n} \cdot \frac{u^n [u + (u-1)\alpha] - u^{\frac{k_{\text{even}}}{2}} \left[u + (u-1) \left(n + \alpha - \frac{k_{\text{even}}}{2} \right) \right]}{(u-1)^2} \\ &\quad (k_{\text{even}} = 2, 4, 6, \dots, 2n) \end{aligned} \quad (7.32)$$

Chapter 7: Stopping Criteria of Delay Evaluation for RLC-Ladder Networks via Lanczos Process

Especially, when substituting $k_{\text{even}} = 2n$ into eqn. (7.32), we obtain $v_{2n} = 0$.

$$\begin{aligned} \mathbf{w} &= \mathbf{A}_L^T \mathbf{w}_1 - \mathbf{w}_1 \alpha_1 - \mathbf{w}_0 \gamma_1 \\ &= \begin{bmatrix} w_1 & w_2 & \cdots & w_{k_{\text{odd}}} & w_{k_{\text{even}}} & \cdots & w_{2n-1} & w_{2n} \end{bmatrix}^T \end{aligned} \quad (7.33)$$

where $w_{k_{\text{odd}}}$ and $w_{k_{\text{even}}}$ are the odd and even components of $\mathbf{w}$, and they are

$$w_{k_{\text{odd}}} = -L \quad (k_{\text{odd}} = 1, 3, 5, \dots, 2n-1) \quad (7.34)$$

$$w_{k_{\text{even}}} = \begin{cases} -\frac{u^{\frac{k_{\text{even}}}{2}-1}}{u-1} \cdot RC & (k_{\text{even}} = 2, 4, 6, \dots, 2n-2) \\ \frac{u^n - (u-1)n-1}{(u-1)^2} \cdot RC & (k_{\text{even}} = 2n) \end{cases} \quad (7.35)$$

Step (4): Setting $i = \frac{k_{\text{odd}}-1}{2}$ and $j = \frac{k_{\text{even}}}{2}$,

$$\rho_2 = \|\mathbf{v}\|_2 \quad (7.36)$$

$$\begin{aligned} &= \frac{C\sqrt{n}}{n} \sqrt{\sum_{i=0}^{n-1} (n+\alpha-i)^2 + \frac{R^2}{(u-1)^4} \sum_{j=1}^n \{u^n [u+(u-1)\alpha] - u^j [u+(u-1)(n+\alpha-j)]\}^2} \\ &= \frac{C\sqrt{n}}{n} \sqrt{\frac{n^3}{3} + \left(\frac{1}{2} + \alpha\right)n^2 + \left(\alpha^2 + \alpha + \frac{1}{6}\right)n + \frac{R^2}{(u-1)^4} \cdot \rho_{2_sub}} \end{aligned}$$

with

$$\begin{aligned} \rho_{2_sub} &= \sum_{j=1}^n \{u^n [u+(u-1)\alpha] - u^j [u+(u-1)(n+\alpha-j)]\}^2 \\ &= (n-1)u^{2n} [u+(u-1)\alpha]^2 \\ &\quad - 2u^n [u+(u-1)\alpha] \frac{[u^2 + (\alpha-n+1)u + n - \alpha]u^n - (u^2 + \alpha u - \alpha + 1)u}{u-1} \\ &\quad + \frac{u^2}{(u+1)^3(u-1)} \cdot \left\{ u^{2n-2} \begin{bmatrix} u^6 + (2\alpha-16)u^5 \\ +(\alpha-8)^2 u^4 \\ +(20-2\alpha)u^3 \\ +(36\alpha-159-2\alpha^2)u^2 \\ +(\alpha-10)^2 \end{bmatrix} - \begin{bmatrix} u^6 + (2\alpha+2)u^5 \\ +(\alpha+1)^2 u^4 \\ +(2-2\alpha)u^3 \\ +(3-2\alpha^2)u^2 \\ +(\alpha-1)^2 \end{bmatrix} \right\} \end{aligned} \quad (7.37)$$

Chapter 7: Stopping Criteria of Delay Evaluation for RLC-Ladder Networks via Lanczos Process

$$\eta_2 = \|\mathbf{w}\|_2, \mathbf{v}_2 = \frac{\mathbf{v}}{\rho_2} \quad \text{and} \quad \mathbf{w}_2 = \frac{\mathbf{w}}{\eta_2} \quad (7.38)$$

For $k = 2$:

$$\text{Step (5): } \delta_2 = \mathbf{w}_2^T \mathbf{v}_2 = \frac{1}{\rho_2 \eta_2} \cdot \mathbf{w}^T \mathbf{v} = \frac{M}{\rho_2 \eta_2} \quad (7.39)$$

where

$$M = \mathbf{w}^T \mathbf{v} \quad (7.40)$$

$$= \frac{-LC(n+2\alpha+1)\sqrt{n}}{2} + \frac{R^2 C^2 \sqrt{n}}{(u-1)^4 (u+1)^2 n} \left\{ \begin{aligned} &u^n (u+1)^2 [u + (u-1)\alpha] (nu - n - u^n + 1) \\ &+ (-u^n \alpha - 2u^n + 2 + n + \alpha) u^3 + \alpha u^n - \alpha u^{2n} \\ &+ (2 + \alpha u^{2n} - \alpha u^n + 2u^{2n} - 4u^n) u^2 \\ &+ (-n - \alpha + \alpha u^n + 1 + u^{2n} - 2u^n) u \end{aligned} \right\}$$

$$\text{Step (6): } \alpha_2 = \frac{\mathbf{w}_2^T \mathbf{A}_L \mathbf{v}_2}{\delta_2} = \frac{\mathbf{w}^T \mathbf{A}_L \mathbf{v}}{M} = \frac{\mathbf{y} \cdot \mathbf{v}}{M} = \frac{Q}{M} \quad (7.41)$$

where

$$\mathbf{y} = \mathbf{w}^T \mathbf{A}_L = [y_1 \quad y_2 \quad \cdots \quad y_{k_{\text{odd}}} \quad y_{k_{\text{even}}} \quad \cdots \quad y_{2n-1} \quad y_{2n}] \quad (7.42)$$

with

$$y_{k_{\text{odd}}} = \frac{LRC}{(u-1)^2} \left[-u^{\frac{k_{\text{odd}}+1}{2}} + (u-1) \left(\frac{k_{\text{odd}}+1}{2} \right) + 1 \right] \quad (k_{\text{odd}} = 1, 3, 5, \dots, 2n-1) \quad (7.43)$$

$$y_{k_{\text{even}}} = \left\{ \begin{aligned} &\frac{-C \left\{ \frac{k_{\text{even}}}{2} (1-u^2) \left[2Lu - L(1+u^2) + R^2 C u^{\frac{k_{\text{even}}}{2}} \right] + R^2 C (u^{k_{\text{even}}+1} - u) \right\}}{2(u+1)(u-1)^3} \\ &\quad (k_{\text{even}} = 2, 4, 6, \dots, 2n-2) \end{aligned} \right. \quad (7.44)$$

Because $v_{2n} = 0$, y_{2n} is useless for obtaining the value of α_2 .

Then setting $i = \frac{k_{\text{odd}}+1}{2}$, $j = \frac{k_{\text{even}}}{2}$ and using eqns. (7.30)-(7.32),

$$Q = \mathbf{y} \cdot \mathbf{v} \quad (7.45)$$

$$= \sum_{k_{\text{odd}}=1,3,\dots,2n-1} (y_{k_{\text{odd}}} v_{k_{\text{odd}}}) + \sum_{k_{\text{even}}=2,4,\dots,2n-2} (y_{k_{\text{even}}} v_{k_{\text{even}}}) + y_{2n} v_{2n}$$

Chapter 7: Stopping Criteria of Delay Evaluation for RLC-Ladder Networks via Lanczos Process

$$\begin{aligned}
 &= \frac{LRC^2\sqrt{n}}{(u-1)^2 n} \sum_{i=1}^n \left\{ \left[-u^i + (u-1)i + 1 \right] (n + \alpha + 1 - i) \right\} \\
 &\quad - \frac{RC^2\sqrt{n}}{2(u+1)(u-1)^5 n} \sum_{j=1}^{n-1} \left\{ \left\{ j(1-u^2) \left[2Lu - L(1+u^2) + R^2Cu^j \right] + R^2C(u^{2j+1} - u) \right\} \right. \\
 &\quad \left. \cdot \left\{ u^n \left[u + (u-1)\alpha \right] - u^j \left[u + (u-1)(n + \alpha - j) \right] \right\} \right\} \\
 \beta_2 &= \frac{\delta_2}{\delta_1} \cdot \eta_2 = \frac{M\sqrt{n}}{\rho_2}
 \end{aligned} \tag{7.46}$$

Step (7): Stop.

Thus, the 2nd-order $\mathbf{T}_q$ ($q = 2$) is obtained and used to approximate $\mathbf{A}_L$. Eqn. (7.2) gives

$$\mathbf{T}_q(q=2) = \begin{bmatrix} \alpha_1 & \beta_2 \\ \rho_2 & \alpha_2 \end{bmatrix} \tag{7.47}$$

In the case of $u=1$, the parameters α_1 , ρ_2 , α_2 and β_2 correspond to eqns. (7.11), (7.19), (7.23) and (7.25), respectively. However, when $u \neq 1$, these parameters correspondingly come from eqns. (7.28), (7.36), (7.41) and (7.46).

Also, from eqn. (7.3), it can be derived that

$$\mathbf{A}_t^T = (-\mathbf{A}_L^{-1})^T \approx (-\mathbf{T}_q^{-1})^T = \hat{\mathbf{A}}_t^T. \tag{7.48}$$

The reduced-order approximation of $\mathbf{A}_t^T$ is attained as the function of $\mathbf{T}_q$ and denoted as $\hat{\mathbf{A}}_t^T$. Thus, the 2nd-order model of $\hat{\mathbf{A}}_t^T$ is computed from eqns. (7.47) and (7.48).

7.2.3 Optimized Global Switching Time Bounds of RLC-Ladder Networks via Lanczos Process

The eigenvalue and corresponding eigenvector of the square matrix $\hat{\mathbf{A}}_t^T$ are denoted as ξ and $\mathbf{x}_\xi$, respectively. They satisfy the following relationship (Varga 1962, Lancaster and Tismenersky 1988):

$$\hat{\mathbf{A}}_t^T \mathbf{x}_\xi = \xi \mathbf{x}_\xi \tag{7.49}$$

Consequently through solving the characteristic equation,

$$\left| \xi \mathbf{E} - \hat{\mathbf{A}}_t^T \right| = 0 \tag{7.50}$$

Chapter 7: Stopping Criteria of Delay Evaluation for RLC-Ladder Networks via Lanczos Process

the eigenvalues of $\hat{\mathbf{A}}_t^T$ can be calculated. According to eqn. (7.48), $\hat{\mathbf{A}}_t^T$ equals $(-\mathbf{T}_q^{-1})^T$ and is a 2nd-order matrix, therefore two eigenvalues of $\hat{\mathbf{A}}_t^T$, ξ_1 and ξ_2 , can be obtained. Here, ξ is the eigenvalue variable and can be substituted by ξ_1 or ξ_2 , i.e.,

$$\xi = \xi_1 \text{ or } \xi = \xi_2 \quad (6.51)$$

After finding the positive eigenvalues ξ_1 and ξ_2 , through solving the eigenequation,

$$(\xi \mathbf{E} - \hat{\mathbf{A}}_t^T) \mathbf{x}_\xi = 0 \quad (7.52)$$

the corresponding eigenvectors are derived as

$$\mathbf{x}_\xi = \begin{bmatrix} x_1(\xi_1) \\ x_2(\xi_1) \end{bmatrix} \quad (\text{corresponds to } \xi_1) \quad (7.53)$$

and

$$\mathbf{x}_\xi = \begin{bmatrix} x_1(\xi_2) \\ x_2(\xi_2) \end{bmatrix} \quad (\text{corresponds to } \xi_2) \quad (7.54)$$

The new bounds of q_{ev} and r_k via Lanczos process (LP) are defined as

$\underline{q}_{ev_LP}$, $\bar{q}_{ev_LP}$, $\underline{r}_{LP}$ and $\bar{r}_{LP}$, which are the same as in eqn. (7.1). They are

$$\underline{q}_{ev_LP} = \min(\xi_1, \xi_2), \quad (7.55)$$

$$\bar{q}_{ev_LP} = \max(\xi_1, \xi_2), \quad (7.56)$$

$$\underline{r}_{LP} = \min(x_1(\xi_1), x_2(\xi_1), x_1(\xi_2), x_2(\xi_2)), \quad (7.57)$$

$$\bar{r}_{LP} = \max(x_1(\xi_1), x_2(\xi_1), x_1(\xi_2), x_2(\xi_2)) \quad (7.58)$$

Then the new bounds of global switching time generated by the optimized method via Lanczos process can be derived from eqn. (5.34) such that

$$\underline{T}_{\lambda_LP} = \max \left\{ 0; -\frac{1}{\bar{q}_{ev_LP}} \ln \frac{\lambda \bar{r}_{LP}}{\underline{r}_{LP}} \right\} \quad \text{and} \quad \bar{T}_{\lambda_LP} = -\frac{1}{\underline{q}_{ev_LP}} \ln \frac{\lambda \underline{r}_{LP}}{\bar{r}_{LP}} \quad (7.59)$$

Late in the chapter, it is shown by examples that all new bounds of q_{ev} , r_k and T_λ via Lanczos process have obvious improvement.

7.2.4 Delay Time Bounds of RLC-Ladder Networks via Lanczos Process

Following similar procedures in Section 6.4, the network delay time bounds can be computed from the GST bounds.

7.3 Numerical Examples and Analysis

The n -cell RLC-ladder networks in Figure 7.1 with a series of diverse cell numbers and u values are selected as examples. Each case has four subclasses corresponding to four different α values (different loads): $\alpha = 0$, $\alpha = n$, $\alpha = 2n$ and $\alpha = 4n$.

Corresponding to the two cases: when $u = 1$ and $u \neq 1$, the related values of R , L and C are set respectively as:

when $u = 1$: $R = 1k\Omega$, $L = 1\mu H$ and $C = 10nF$;

when $u \neq 1$: $R = 1\Omega$, $L = 1nH$ and $C = 10pF$.

For all networks, the extended Marinov method in Section 5.3.5 and the optimized method via Lanczos process are used to compute their delay bounds. Then the delay bounds obtained are compared with the exact delays.

For three networks, Tables 7.1-7.3 list the comparison of the bounds of eigenvalue q_{ev} and eigenvector component r_k produced by the two methods. The delay bounds using the two methods and the exact delay are computed and compared in Tables 7.4-7.12, where the bound width ε_D is defined by

$$\varepsilon_D = (\bar{T}_D - \underline{T}_D) / T_D \quad (7.60)$$

where $\underline{T}_D$ is the lower delay bound, $\bar{T}_D$ is the upper delay bound and T_D is the exact delay.

In all the tables, the underlined bounds are produced by the method in Section 5.3.5, and the other bounds are generated by the method via Lanczos process.

Chapter 7: Stopping Criteria of Delay Evaluation for RLC-Ladder Networks via Lanczos Process

Table 7.1 Comparison of eigenvalue and eigenvector bounds for RLC-ladder network with $u = 1$ and $n = 30$

α	Bound Esti Method	$\underline{q}_{ev}$	$\bar{q}_{ev}$	$\underline{r}$	$\bar{r}$
0	Via Lanczos	2.65414203e+2	1.74124261e+3	0.40183833	0.91571063
	Ex-Manrinov	2.15039890e+2	1.00000000e+6	2.15039890e-6	1
30	Via Lanczos	81.24913612	1.08266253e+3	0.18741586	0.98228066
	Ex-Manrinov	73.09941520	1.00000000e+6	7.30994152e-7	1
60	Via Lanczos	47.08198185	1.02153269e+3	0.11584150	0.99326771
	Ex-Manrinov	44.09171076	1.00000000e+6	4.40917107e-7	1
120	Via Lanczos	25.51121170	9.87852768e+2	0.06518292	0.99787333
	Ex-Manrinov	24.58210423	8.26446281e+5	2.45821042e-7	1

Table 7.2 Comparison of eigenvalue and eigenvector bounds for RLC-ladder network with $u = 5$ and $n = 15$

α	Bound Esti Method	$\underline{q}_{ev}$	$\bar{q}_{ev}$	$\underline{r}$	$\bar{r}$
0	Via Lanczos	12.48992958	77.99103564	0.16897112	0.98562100
	Ex-Manrinov	10.48575999	6.24999999e+9	1.04857599e-10	1
15	Via Lanczos	0.81704978	75.17264196	0.01347987	0.99990914
	Ex-Manrinov	0.80659691	6.24999999e+9	8.06596913e-12	1
30	Via Lanczos	0.42224229	75.09524594	0.00701311	0.99997541
	Ex-Manrinov	0.41943040	3.22580645e+9	4.19430397e-12	1
60	Via Lanczos	0.21472506	75.05494420	0.00357899	0.99999360
	Ex-Manrinov	0.21399510	1.63934426e+9	2.13995101e-12	1

Table 7.3 Comparison of eigenvalue and eigenvector bounds for RLC-ladder network with $u = 10$ and $n = 10$

α	Bound Esti Method	$\underline{q}_{ev}$	$\bar{q}_{ev}$	$\underline{r}$	$\bar{r}$
0	Via Lanczos	89.01266255	9.89000309e+2	0.09665588	0.99531786
	Ex-Manrinov	80.99999942	9.09090909e+9	8.09999994e-10	1
10	Via Lanczos	8.17424947	9.80034582e+2	0.00970030	0.99995295
	Ex-Manrinov	8.09999934	9.09090909e+9	8.09999935e-11	1
20	Via Lanczos	4.28364489	9.79643978e+2	0.00510484	0.99998697
	Ex-Manrinov	4.26315771	4.76190476e+9	4.26315771e-11	1
40	Via Lanczos	2.19458008	9.79435641e+2	0.00262121	0.99999656
	Ex-Manrinov	2.18918914	2.43902439e+9	2.18918914e-11	1

Chapter 7: Stopping Criteria of Delay Evaluation for *RLC-Ladder Networks* via *Lanczos Process*

Table 7.4 Comparison of delay bounds for *RLC-ladder network* with $u = 1$ and $n = 20$

α	λ_D	T_D (s)	Bound Esti Method	$\underline{T}_D$ (s)	$\overline{T}_D$ (s)	ε_D
0	0.32643089	0.00159033	Via Lanczos	8.054409e-5	0.00328358	2.0141
			Ex-Manrinov	0	0.02808911	17.6624
20	0.27879074	0.00442326	Via Lanczos	0	0.01609921	3.6397
			Ex-Manrinov	0	0.08936290	20.2029
40	0.27166201	0.00720089	Via Lanczos	0	0.03261155	4.5288
			Ex-Manrinov	0	0.15312203	21.2643
80	0.26737404	0.01274817	Via Lanczos	0	0.07046964	5.5278
			Ex-Manrinov	0	0.28501036	22.3570

Table 7.5 Comparison of delay bounds for *RLC-ladder network* with $u = 1$ and $n = 25$

α	λ_D	T_D (s)	Bound Esti Method	$\underline{T}_D$ (s)	$\overline{T}_D$ (s)	ε_D
0	0.32480937	0.00246143	Via Lanczos	1.231553e-4	0.00511567	2.0283
			Ex-Manrinov	0	0.04490602	18.2439
25	0.27636927	0.00688979	Via Lanczos	0	0.02518186	3.6550
			Ex-Manrinov	0	0.14337852	20.8103
50	0.26919053	0.01122994	Via Lanczos	0	0.05106584	4.5473
			Ex-Manrinov	0	0.24583267	21.8908
100	0.26488422	0.01989761	Via Lanczos	0	0.11040425	5.5486
			Ex-Manrinov	0	0.45758467	22.9970

Table 7.6 Comparison of delay bounds for *RLC-ladder network* with $u = 1$ and $n = 30$

α	λ_D	T_D (s)	Bound Esti Method	$\underline{T}_D$ (s)	$\overline{T}_D$ (s)	ε_D
0	0.32373191	0.00352190	Via Lanczos	1.746965e-4	0.00735262	2.0381
			Ex-Manrinov	0	0.06593054	18.7202
30	0.27475342	0.00990058	Via Lanczos	0	0.03628873	3.6653
			Ex-Manrinov	0	0.21095575	21.3074
60	0.26754208	0.01615050	Via Lanczos	0	0.07364295	4.5598
			Ex-Manrinov	0	0.36181149	22.4025
120	0.26322379	0.02863197	Via Lanczos	0	0.15927035	5.5627
			Ex-Manrinov	0	0.67339283	23.5189

Chapter 7: Stopping Criteria of Delay Evaluation for RLC -Ladder Networks via Lanczos Process

Table 7.7 Comparison of delay bounds for RLC -ladder network with $u = 5$ and $n = 13$

α	λ_D	$T_D(s)$	Bound Esti Method	$\underline{T}_D(s)$	$\bar{T}_D(s)$	ε_D
0	0.05025845	0.00287073	Via Lanczos	6.349618e-4	0.01519022	5.0702
			Ex-Manrinov	0	0.08678482	30.2309
13	0.04824336	0.03034057	Via Lanczos	0	0.30846529	10.1668
			Ex-Manrinov	0	1.09695864	36.1548
26	0.04813904	0.05783792	Via Lanczos	0	0.64767881	11.1982
			Ex-Manrinov	0	2.15178518	37.2037
52	0.04810850	0.11283456	Via Lanczos	0	1.37891446	12.2207
			Ex-Manrinov	0	4.31383746	38.2315

Table 7.8 Comparison of delay bounds for RLC -ladder network with $u = 5$ and $n = 15$

α	λ_D	$T_D(s)$	Bound Esti Method	$\underline{T}_D(s)$	$\bar{T}_D(s)$	ε_D
0	0.04355743	0.07176837	Via Lanczos	0.01756780	0.39209340	5.2185
			Ex-Manrinov	0	2.49024323	34.6983
15	0.04175561	0.86427087	Via Lanczos	0	9.15781225	10.5960
			Ex-Manrinov	0	35.60550279	41.1972
30	0.04171352	1.65747269	Via Lanczos	0	19.27064056	11.6265
			Ex-Manrinov	0	70.03360719	42.2532
60	0.04169175	3.24392186	Via Lanczos	0	41.02977255	12.6482
			Ex-Manrinov	0	1.404130e+2	43.2849

Table 7.9 Comparison of delay bounds for RLC -ladder network with $u = 5$ and $n = 18$

α	λ_D	$T_D(s)$	Bound Esti Method	$\underline{T}_D(s)$	$\bar{T}_D(s)$	ε_D
0	0.03629079	8.97104891	Via Lanczos	2.46982988	50.95491070	5.4046
			Ex-Manrinov	0	3.710142e+2	41.3568
18	0.03489229	1.27863453e+2	Via Lanczos	0	1.423792e+3	11.1353
			Ex-Manrinov	0	6.222814e+3	48.6677
36	0.03469629	2.46847915e+2	Via Lanczos	0	3.004684e+3	12.1722
			Ex-Manrinov	0	1.227806e+4	49.7394
72	0.03468011	4.84820088e+2	Via Lanczos	0	6.396057e+3	13.1926
			Ex-Manrinov	0	2.461682e+4	50.7752

Chapter 7: Stopping Criteria of Delay Evaluation for RLC-Ladder Networks via Lanczos Process

Table 7.10 Comparison of delay bounds for RLC-ladder network with $u = 10$ and $n = 10$

α	λ_D	T_D (s)	Bound Esti Method	$\underline{T}_D$ (s)	$\overline{T}_D$ (s)	ε_D
0	0.05613240	0.00894584	Via Lanczos	5.542333e-4	0.05855287	6.4833
			Ex-Manrinov	0	0.29400036	32.8645
10	0.05559620	0.08592036	Via Lanczos	0	0.92059726	10.7145
			Ex-Manrinov	0	3.22545859	37.5401
20	0.05557410	0.16293447	Via Lanczos	0	1.90669209	11.7022
			Ex-Manrinov	0	6.27902266	38.5371
40	0.05556200	0.31696380	Via Lanczos	0	4.02554107	12.7003
			Ex-Manrinov	0	12.53211066	39.5380

Table 7.11 Comparison of delay bounds for RLC-ladder network with $u = 10$ and $n = 12$

α	λ_D	T_D (s)	Bound Esti Method	$\underline{T}_D$ (s)	$\overline{T}_D$ (s)	ε_D
0	0.04677708	0.89457384	Via Lanczos	0.07249631	6.07524235	6.7102
			Ex-Manrinov	0	35.31051719	39.4719
12	0.04632400	10.13226129	Via Lanczos	0	1.139800e+2	11.2492
			Ex-Manrinov	0	4.527609e+2	44.6851
24	0.04627167	19.37434483	Via Lanczos	0	2.372011e+2	12.2430
			Ex-Manrinov	0	8.853154e+2	45.6952
48	0.04634933	37.85792308	Via Lanczos	0	5.012591e+2	13.2405
			Ex-Manrinov	0	1.767969e+3	46.7001

Table 7.12 Comparison of delay bounds for RLC-ladder network with $u = 10$ and $n = 13$

α	λ_D	T_D (s)	Bound Esti Method	$\underline{T}_D$ (s)	$\overline{T}_D$ (s)	ε_D
0	0.04317123	8.94566712	Via Lanczos	0.80093297	61.71075904	6.8089
			Ex-Manrinov	0	<u>3.825225e+2</u>	<u>42.7606</u>
13	0.04278662	1.09025870e+2	Via Lanczos	0	1.251896e+3	11.4826
			Ex-Manrinov	0	<u>5.257937e+3</u>	<u>48.2265</u>
26	0.04293692	2.09145669e+2	Via Lanczos	0	2.608443e+3	12.4719
			Ex-Manrinov	0	<u>1.029751e+4</u>	<u>49.2361</u>
52	0.04273885	4.09386947e+2	Via Lanczos	0	5.518130e+3	13.4790
			Ex-Manrinov	0	<u>2.057253e+4</u>	<u>50.2521</u>

Chapter 7: Stopping Criteria of Delay Evaluation for RLC-Ladder Networks via Lanczos Process

From Tables 7.1-7.3, it can be seen that, using the method via Lanczos process, the bounds of eigenvalue and eigenvector component are much tighter than the bounds from the method in Section 5.3.5.

In Tables 7.4-7.12, it can be concluded that the delay bounds from the method via Lanczos process are much tighter than those from the other method. Comparing with the exact delay, it is obvious that the delay bounds via Lanczos process are greatly improved. This fact is also evident from the comparison of the bound width ε_D .

7.4 Summary

Early analysis of ICs modeled interconnects as lumped or distributed *RC*-cell interconnects. However, as a result of longer interconnect wires and an increase in the number of layers of interconnect, the parasitic modeling of on-chip interconnect has progressively changed over the years. Furthermore, the use of increasingly higher frequency of operation implies that the inductive effects of interconnects can no longer be ignored, and on-chip interconnects are best modeled as a network of coupled lines. Hence, their delay estimation has to consider the effect of parasitic coupling between the lines, as the noise due to parasitic coupling between signal wires can result in functional failure or logic error by introducing delays into the circuit. If accurate interconnect delay estimation is to be achieved, modeling interconnects as *RLC*-cell interconnects is necessary.

As the fundamental structure in *RLC*-cell interconnects, the analysis to the *RLC*-ladder network has obvious importance. Chapter 6 proposed the method via Lanczos process to compute the optimized delay time bounds of *RC*-ladder networks. The method is extended to the *RLC*-ladder network in this chapter. Thus, the delay bounds for the *RLC*-ladder network are achieved. Compared with the exact delay and the delay bounds obtained in Chapter 5, the delay bounds of *RLC*-ladder networks from the optimized method via Lanczos process are much tighter and more accurate.

Chapter 8: Conclusions and Future Work

8.1 Conclusions

In this thesis, problems, advantages, existing and new methods in the field of timing analysis of interconnect networks have been investigated. The research includes exact and approximate delay time computation, delay time bound derivations using a number of methods, and numerical evaluations of various examples to verify theoretical formulation. Because of its conventional usage, the step-response method is applied to characterize the delay time of interconnect networks.

The wave-chain (T-matrix) and the state-space methods have been used to compute the exact delay of an interconnect network. The exact values are mainly used as reference and for comparisons. The AWE and Padé approaches have been used to compute the approximate delay. The technique by Marinov and Shivahumar (1996) and the Lanczos process have been used to derive the delay bounds.

In this research work, the following contributions have been made:

(1) Evaluating Various Approximation Methods to Compute Delay

Two approximation methods, the AWE technique and Padé approximation, have been investigated for approximating an actual system behaviour with a lower order model and for computing its approximate system delay time. Conventional circuits like Butterworth, Chebyshev and Bessel low-pass filters, and *RC*-tree networks have been used to verify the effectiveness of AWE in computing delay. Indeed, the AWE has been found to be very efficient in evaluating the delay values of high-order interconnect networks and with good accuracy, but at the same time the AWE suffers from numerical instability inherent from Padé approximation. Nevertheless, the inadequacies of AWE can be complemented by well-known remedial procedures, or by the pole interpolation method to be discussed in the next sub-section.

For interconnect networks containing lossy transmission lines, Shao and Chen (2000)

describes a two-pole method in computing approximate dominant poles and residues. In this thesis, the method is extended to include more than one pair of approximate poles so that the accuracy of computation can be increased.

(2) Improved AWE-Based Delay Calculation by Pole Interpolation

This interpolation method has not been discussed in any literature. The AWE employs Padé approximation (PA) for its moment matching. Since PA is very sensitive to moments, small round-off error in the moments can cause some generated approximate poles to be unstable. However, the remaining acceptable poles generally do not show any sign of outlier phenomena. This implies a way to replace those unstable poles. In this thesis we have shown that the real parts of those unstable poles can be efficiently and approximately corrected, and moved to the LHP by interpolation. The inclusion of those restored poles generally improves the overall delay estimation of the circuit without incurring significant computational time.

(3) Extension of Delay Bound Computation Method

Marinov and Shivakumar (1996) have presented the GST bounds for *RC*-ladder networks, a basic model in integrated circuits. Since *RC*-tree, *RLC*-ladder and *RLC*-tree networks are also widely used in integrated circuits, similar GST bounds are established in this thesis for *RC*-tree, *RLC*-ladder and *RLC*-tree networks. Moreover, the delay time bounds are computed from the GST bounds for the above mentioned networks. It is also found that the delay time bounds become wider with the increase of load.

(4) Optimized Delay Bounds via Lanczos Process

A novel method to derive the delay time bounds of *RC*-ladder networks using Lanczos Process (LP) is presented. In this method, the Lanczos algorithm is applied to find the reduced-order model of the state matrix of the network. Then an arbitrary order of *RC*-ladder networks can be generated from the reduced-order model of the state matrix with little numerical degradation, and the network performance can be described well using only a 2nd-order square matrix model. The GST and delay bounds of networks can then be obtained.

Numerical results show that the GST and delay bounds via Lanczos process are much tighter than the bounds by Marinov and Shivakumar (1996) and by Hou *et.al.* (2000). The corresponding improvement proof is also given. The method is also applied to a *RLC*-ladder network to obtain its delay bounds.

In summary, this project has achieved the following:

1. Different model-reduction methods have been evaluated. A two-pole method is extended to increase the accuracy of approximation method.
2. An interpolation method has been proposed so that interpolated AWE poles can fill the gaps of missing poles and increase accuracy in evaluating delay.
3. A delay bound method for *RC*-ladder networks has been extended for *RC*-tree, *RLC*-ladder and *RLC*-tree networks.
4. A novel method of deriving delay time bounds for *RC*- and *RLC*-ladder networks via Lanczos process has been proposed and verified.

8.2 Suggestions for Future Work

8.2.1 Timing Analysis of Wave-Digital Interconnect Networks

Wave-digital filters (WDFs) are a special class of recursive digital filters with same rather remarkable properties (Fettweis 1986). The structures of WDFs are derived from classical resistance-terminated lossless analogue filter networks by making use of the familiar wave variables of classical network theory and microwave networks together with relevant scattering matrices and a series of rather straightforward bilinear transformations. A key element in WDF technology is the so-called adaptor, an n -part module in which all the additions and multiplications take place. An adaptor forms the digital representation of Kirchhoff's laws in classical network theory. The techniques developed for computing delay in this project can be similarly applied to wave-digital interconnect networks.

8.2.2 Timing Analysis of Computer Networks

The methods applied to compute approximate delay and delay bounds of interconnect networks may also be applied to computer networks which also contain similar interconnection structure. If a computer or communication network can be converted to a *RC*- or *RLC*-equivalent, then the methods explored in this project can be applied to such a network for the purpose of studying the network delay and related issues such as the quality of service.

Same as the throughput, the delay is one of the most important parameters of computer networks which can help us to describe and understand the performance of the network (Leon-Garcia and Widjaja 2000). The delay is defined as the elapsed time between when a message is sent by the source and when it is received by the destination. Delay consists of queuing delay (waiting time), transmission delay and propagation delay. Based on Internet Protocol (IP), a message can be divided into many packets. There exists a path for transmitting a packet from the source to the destination, and the internetworking devices on the path can be regarded as queuing systems. Every internetworking device relays the packets and will result in a relay delay that is a combination of queuing delay and transmission delay. Neglecting propagation delay, the end-to-end delay of a packet is equal to the sum of relay delays of the internetworking devices along the path.

Computer network is an interconnected collection of autonomous computers and lots of internetworking devices, such as repeater, hub, switch, bridge, router, gateway, and links of various media. Based on the properties of internetworking devices and links, computer network can be converted into the equivalent electrical network. Because modern large area network such as Internet contains countless devices, computers and transmission links, it is difficult to estimate the delay of a message accurately. Although some simulator softwares such as COMNET and OPNET (Hoover and Perry 1989, Law and Kelton 1999) are used to approximate the behaviors of an actual system, the analysis is very time-consuming. If a computer network can be modeled as a *RC*- or *RLC*-interconnect, the approximate delay computation methods and delay bound methods can be applied to speed up computation and analysis.

In the following, the equivalent circuits of three fundamental communication network structures are discussed briefly:

(1) Equivalent Electrical Network of Internetworking Devices

The delay function of internetworking devices is similar to that of energy-storage elements in electrical circuits. Thus, a computer network containing computers, internetworking devices and links can be modeled as an equivalent electrical interconnect network. Capacitors and inductors are used to simulate the devices resulting in data transmission delay. An Ethernet Local Area Network (LAN) with star topology shown in Figure 8.1, an Ethernet LAN with tree topology shown in Figure 8.2, a backbone network with mesh topology shown in Figure 8.3 and a token ring LAN shown in Figure 8.4 are four basic examples (Oppenheimer 1999, Stallings 1999, Wang 1999, Stallings 2000). For simplicity, the effect of links is neglected.

Figure 8.1 An Ethernet LAN with star topology

Figure 8.2 An Ethernet LAN with tree topology

Figure 8.3 A backbone network with mesh topology

Figure 8.4 A token ring LAN

In Figure 8.1, the transmission path from file server to station K is $H1 \rightarrow H2 \rightarrow H3$, the equivalent electrical interconnect network is shown in Figure 8.5. In Figure 8.2, the transmission path from station A to station E is $B1 \rightarrow B3$, the equivalent electrical interconnect network is shown in Figure 8.6. Observing Figure 8.3, the transmission path from station A to station C is $R1 \rightarrow R4 \rightarrow R6$, and the one from station A to station D is $R1 \rightarrow R2 \rightarrow R3 \rightarrow R5$. Two equivalent electrical interconnect networks in Figure 8.7 are given. For the token ring LAN in Figure 8.4, the transmission path from station A to station D is $R1 \rightarrow R2 \rightarrow R3 \rightarrow R4 \rightarrow R5$, and its equivalent electrical network is shown in Figure 8.8. In the modeling process, a floating resistor and a capacitor connected to the earth are used to simulate the relay delay effect of an internetworking device.

Equivalent electrical network of the path:
H1→H2→H3 (Server →Station K)

Figure 8.5 Equivalent electrical network of the star network

Equivalent electrical network of the path:
B1→ B3 (Station A →Station E)

Figure 8.6 Equivalent electrical network of the tree network

Equivalent electrical network of the path: R1→ R4→ R6

Equivalent electrical network of the path: R1→R2→R3→R5

Figure 8.7 Equivalent electrical network of the mesh network

Equivalent electrical network of the path:
R1→R2→R3→R4→R5 (Station A →Station D)

Figure 8.8 Equivalent electrical network of the token ring LAN

Figure 8.9 A WAN consisting of many network segments with different topologies

Applying the model method shown in Figures 8.1 to 8.4 to a larger interconnect network, the equivalent electrical network obtained has a similar structure. The network shown in Figure 8.9 is a Wide Area Network (WAN) (Sloman 1994) in which LAN A is connected with LAN B through the backbone network. In LAN A, segment A3 is a bus-topology Ethernet, and segment A4 is an Asynchronous Transfer Mode (ATM) network. In LAN B, segment B1 and B3 are two bus-topology Ethernets, segment B2 is a token ring network, and segment B4 is a network with two-level star topology. Station A is one station in LAN segment A4, station B is one station in LAN segment B3, and station C is one

station in LAN segment B4. Assuming station A is the source point and station B as well as C are two destination points, the transmission path from station A to station B is $S_ATM \rightarrow LAD \rightarrow B1 \rightarrow G1 \rightarrow R1 \rightarrow R2 \rightarrow R3 \rightarrow R5 \rightarrow G2 \rightarrow R7 \rightarrow R8 \rightarrow Re1 \rightarrow Re2 \rightarrow Re3 \rightarrow Re4 \rightarrow B2$, and the transmission path from station A to station C is $S_ATM \rightarrow LAD \rightarrow B1 \rightarrow G1 \rightarrow R1 \rightarrow R2 \rightarrow R3 \rightarrow R5 \rightarrow G2 \rightarrow B3 \rightarrow B4 \rightarrow H1 \rightarrow H2$.

The equivalent electrical network configurations of these two paths are shown in Figure 8.10. It is noticed that they are two connected RC-ladder networks.

Figure 8.10 Equivalent electrical network of the WAN

(2) Equivalent Electrical Model of Long Distance Links

In the above example, the effect of links is neglected for simplicity. However, in reality, the propagation delay caused by links between devices is a very important component of delay. Especially, some long distance transmission links, such as the links between R7 and R8 or between B3 and B4, contribute greatly to the propagation delay. Thus, long distance transmission links can be modeled as lossy transmission lines to reflect their effect of inductance. Section 3.5.1 has already represented equivalent circuits of transmission lines in details. In this way, long distance links can also be modeled as electrical circuit in internetworking devices.

The network shown in Figure 8.11 is a simple example, and its equivalent electrical model is shown in Figure 8.12. The example is a large area LAN consisting of 4 long distance transmission links. Assuming that station A sends a message to station B, the transmission path is $R2 \rightarrow \text{LDTL 1} \rightarrow R1 \rightarrow \text{Re1} \rightarrow \text{LDTL 2} \rightarrow \text{Re2} \rightarrow B1 \rightarrow \text{LDTL 3} \rightarrow B2 \rightarrow \text{Re3} \rightarrow \text{LDTL 4} \rightarrow \text{Re4}$. In equivalent circuit, one can use *RC*- and *RLC*-segment to approximate the behaviors of internetworking devices and long distance transmission links respectively.

Figure 8.11 A large LAN containing 4 long distance transmission links

Figure 8.12 Equivalent electrical network of LAN including long distance transmission links

(3) Equivalent Electrical Model of Bifurcated Routing Network

The shortest path routing which looks for the path with minimum distance is used, and it is most efficient to low load networks. However, the shortest path routing has two drawbacks (Spragins 2000): the first is that it uses only one path per Origin/Destination (OD) pair and potentially limiting the throughput of a network; the second is its limited capability to adapt to changing traffic conditions. So, the optimal routing (Kershenbaum 1999) is used to eliminate the above drawbacks by splitting OD pair traffic at strategic points and shifting traffic gradually between alternative paths. The optimal routing is based on sophisticated mathematical theory of multi-commodity flow and widely used in practice.

The example shown in Figure 8.13 explains why optimal routing is needed. All links have a capacity of 10 units, say, single destination is station B, single origin is station A, and the offered load is equal to 10 units. The network in Figure 8.13 is the same as the backbone network in Figure 8.9. It is known that the shortest path from station A to station B is $R1 \rightarrow R2 \rightarrow R3 \rightarrow R5$. However, because of the limited capacity of every link, using only

one path will result in large delays. Hence, two paths are used to transmit the message from station A. Another path is $R1 \rightarrow R4 \rightarrow R6 \rightarrow R5$. In the same principle, the equivalent electrical interconnect network is shown in Figure 8.14. It can be seen that the equivalent electrical network is a RC -tree interconnect with two branches.

Figure 8.13 Bifurcated routing network

Equivalent electrical network of the path:
1. $R1 \rightarrow R2 \rightarrow R3 \rightarrow R5$
2. $R1 \rightarrow R4 \rightarrow R6 \rightarrow R5$

Figure 8.14 Equivalent electrical network of the bifurcated routing network

List of Publications

Conferences

- Cao, Y. and Tan, E. C., 2002, "Delay computation of large interconnect networks," *Proceedings of the International Conference on Scientific & Engineering Computation (IC-SEC 2002)*, Dec. 3-5, Singapore, pp.227-232.
- Cao, Y. and Tan, E. C., 2003, "Interpolation of poles in Asymptotic Waveform Evaluation," *Proceedings of the Fourth International Conference on Information, Communications & Signal Processing and Fourth Pacific-Rim Conference on Multimedia (IEEE ICICS-PCM 2003)*, Dec.15-18, Singapore, CD-ROM.
- Cao, Y. and Tan, E. C., 2004, "Delay-time bounds for RC-tree networks," *Proceedings of the 10th International Symposium on Integrated Circuits Devices & Systems on Silicon (IEEE ISIC2004)*, Sep. 8-10, Singapore, CD-ROM.
- Cao, Y. and Tan, E. C., 2004, "Two-pole method extension in delay estimation," *Proceedings of the 10th International Symposium on Integrated Circuits Devices & Systems on Silicon (IEEE ISIC2004)*, Sep. 8-10, Singapore, CD-ROM.
- Cao, Y. and Tan, E. C., 2004, "Stopping criteria for evaluating delays in RLC-tree networks," *IEEE Analog and Digital Techniques in Electrical Engineering (IEEE TENCON 2004)*, Nov. 21-24, Thailand, CD-ROM.

Journals

- Cao, Y. and Tan, E. C., "Improving AWE-based delay computation by pole interpolation," (Submitted for possible publication).
- "Delay time bounds for RC-tree networks," (Manuscript in preparation).
- "Delay time bounds for RLC-tree networks," (Manuscript in preparation).
- "Optimized delay bounds of RC-ladder networks via Lanczos process," (Manuscript in preparation).
- "Stopping criteria of delay evaluation for RLC-ladder networks via Lanczos process," (Manuscript in preparation).

References

- Alaybeyi, M. M., Lee, J. Y. and Rohrer, R. A., Nov. 1992, "Numerical integration algorithms and asymptotic waveform evaluation (AWE)," *Proceedings of Tech. Dig. IEEE/ACM International Conference on Computer-Aided Design*, pp.76-79.
- Anastakis, D. F., Gopal, N., Kim, S. Y. and Pillage L. T., Jun. 1993, "On the stability of approximations in asymptotic waveform evaluation," *Proceedings of the 29th ACM/IEEE Design Automation Conference*, pp.207-212.
- Anderson, E. M., 1985, *Electric Transmission Line Fundamentals*, Reston, VA: Reston Publishing Company, Inc., pp.229-255.
- Arora, N., Raol, K. V., Schumann, R. and Richardson L. M., Jan. 1996, "Modeling and exaction of interconnect capacitances for multilayer VLSI circuits," *IEEE Trans. Computer Aided Design*, vol.15, pp.58-67.
- Baker, G. A. and Graves-Morris, P., 1996, *Padé Approximants-Encyclopedia of Mathematics and its Applications*, London: Cambridge University Press, vol.59.
- Bakoglu, H. B., 1990, *Circuits, Interconnections, and Pakaging for VLSI*, Reading, MA: Addison-Wesley Publishing Company, pp. 243-250.
- Carlson, A. B., 1996, *Circuit: Engineering Concepts and Analysis of Linear Electric Circuits*, New York, NY: John Wiley & Sons, Inc., pp.695-725.
- Celik, M., Pileggi, L. and Odabasioglu, A., 2002, *IC Interconnect Analysis*, Norwell, Massachusetts: Kluwer Academic Publishers, pp.25-57.
- Chen, C. P., Chen, Y. P. and Wong, D. F., 1996, "Optimal wire-sizing formula under the Elmore delay model," *Proceedings of the 33rd ACM/IEEE Design Automation conference*, pp. 487-490.
- Chen, C. P. and Wong, D. F., 1996, "A fast algorithm for the optimal wire-sizing under the Elmore delay model," *Proceedings of IEEE ISCAS*, vol. 4, pp. 412-415.
- Chen, C. P. and Wong, D. F., 1997, "Optimal wire-sizing function with fringing capacitance consideration," *Proceeding of ACM/IEEE Design Automation conf.*, pp. 604-607.

References

-
- Chiprout, E. and Nakhla, M. S., 1994, *Asymptotic Waveform Evaluation and Moment Matching for Interconnect Analysis*, Boston, MA: Kluwer Academic Publishers, pp. 243-281.
- Chirlian, P. M., 1969, *Basic Network Theory*, New York, NY: McGraw-Hill Book Company, pp.486-491.
- Choudry, U. and Sangiovanni-Vincentelli, A., Apr. 1995, "Automatic generation of analytical models for interconnect capacitances," *IEEE Trans. Computer Aided Design*, vol.14, no.4, pp.470-480.
- Comer, D. E., 2000, *Internetworking With TCP/IP Volume 1: Principles, Protocols and Architecture, 4th Edition*, New York, N.Y.: McGraw-Hill, pp. 78-102.
- Etter, D. M., 1997, *Engineering Problem Solving with MATLAB, Second Edition*, Englewood Cliffs, NJ: Prentice-Hall, Inc.
- Fausett, L. V., 1999, *Applied Numerical Analysis using Matlab*, Englewood Cliffs, NJ: Prentice-Hall, Inc., pp. 119-172.
- Feldmann, P. and Freund, R. W., May. 1995, "Efficient linear circuit analysis by Padé approximation via the Lanczos process," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol.14, no.5, pp.639-649.
- Fettweis, A., Feb. 1986, "Wave digital filter: theory and practice," *Proceedings of IEEE*, vol.74, no.2, pp.270-327.
- Fishburn J. P. and Schevon, C. A., Dec. 1995, "Shaping a distributed-RC line to minimize Elmore delay", *IEEE Trans. Circuits and Systems-I: Fundamental Theory and Applications*, vol, 42, no. 12, pp. 1020-1022.
- Freund, R. W., 1993, "The look-ahead Lanczos process for large non-symmetric matrices and related algorithms," *Linear Algebra for Large Scale and Real-Time Applications*, Dordrecht, The Netherlands: Kluwer Academic Publishers, pp.137-163.
- Frye, R. C. and Chen, H. Z., Nov. 1992, "Optimal self-damped lossy transmission line interconnections for multichip modules," *IEEE Trans. Circuits and Systems II: Analog and Digital Signal Processing*, vol.39, no.11, pp.765-771.

References

-
- Gardiol, F. E., 1987, *Lossy Transmission Lines*, Norwood, MA: Artech House, Inc., pp.146-164.
- Gerald, C. F. and Wheatley, P. O., 1994, *Applied Numerical Analysis*, Reading, MA: Addison-Wesley Publishing Company, pp. 265-312.
- Golub, G. E., 1996, *Matrix Computations*, Baltimore, MD: Johns Hopkins University Press, pp. 321-334.
- Gupta, R., Tutuianu, B. and Pileggi, L., Jan. 1997, "The Elmore delay as a bound for RC trees generalized input signals", *IEEE Trans. Computer-Aided Design*, vol.16, no.1, pp.95-104.
- Hanchate, N. and Ranganathan, N., Jan. 2006, "A linear time algorithm for wire sizing with simultaneous optimization of interconnect delay and crosstalk noise", *Proceedings of the 19th International Conference on VLSI Design (VLSID'06)*, pp.283-290.
- Hou, C., Gao, F. and Qian, J., Feb. 2000, "Improved delay time estimation of RC ladder networks", *IEEE Trans. Circuits Systems I: Fundamental Theory and Applications*, vol.47, no.2, pp.242-246.
- Hoover, S. V. and Perry, R. F., 1989, *Simulation—A Problem Solving Approach*, Boston, MA: Addison-Wesley Longman Publishing Company, pp. 225-279.
- Huang, X., 1990, *Padé Approximation of Linear(ized) Circuit Responses*, Ph.D. dissertation, Dept. Electrical and Computer Eng., Pittsburgh, PA: Carnegie Mellon University.
- Huang, X., Raghavan, V. and Rohrer, R. A., Nov. 1990, "AWEsim: a program for the efficient analysis of linear(ized) circuits," *Proceedings of Tech. Dig. IEEE/ACM International Conference on Computer-Aided Design*, pp.534-537.
- Yu, Q. and Kuh, E. S. May 2001, "Moment computation of lumped and distributed coupled RC trees with application to delay and crosstalk estimation", *Proceedings of IEEE*, vol. 89, no. 5, pp. 772-788.
- Kahng, A. B. and Muddu, S., Sep. 1994a, "Optimal equivalent circuits for interconnect delay calculations using moments," *Proceedings of the European Design Automation Conference (IEEE)*, pp.164-169.

References

-
-
- Kahng, A. B. and Muddu, S., Feb. 1994b, "Two-pole analysis of interconnect tree," *Proceedings of the Multi-Chip Module Conference (IEEE)*, pp.105-110.
- Kershenbaum, A., 1999, *Telecommunication Network Design Algorithms*, Boston, MA: McGraw-Hill Book Company, pp. 354-380.
- Kuo, B. C., 1995, *Automatic Control Systems*, 7th edn, Englewood Cliffs, NJ: Prentice-Hall, Inc., pp. 65-98.
- Kuo, F. F., 1966, *Network Analysis and Synthesis*, New York, NY: John Wiley & Sons, Inc, pp.368-395.
- Lam, H. Y. F., 1979, *Analog and Digital Filters Design and Realization*, Englewood Cliffs, NJ: Prentice-Hall, Inc., pp.221-321.
- Lancaster, P. and Tismenersky, M., 1988, *The Theory of Matrices, Second Edition, with Applications*, New York, NY: John Wiley & Sons, Inc., pp. 465-497.
- Law, A. M. and Kelton, W. D., 1999, *Simulation Modeling & Analysis*, Boston, MA: McGraw-Hill Book Company, pp. 119-137.
- Leon-Garcia, A. and Widjaja, I., 2000, *Communication Networks: Fundamental Concepts and Key Architectures*, Boston, MA: McGraw-Hill Book Company, pp. 72-83.
- Marinov, C. A., Jan. 1998, "Transient bounding in networks with distributed and lumped parameters," *IEEE Trans. Circuits Systems I: Fundamental Theory and Applications*, vol.45, no.1, pp.11-25.
- Marinov, C. A. and Neittaanm, P., 1991, *Mathematical Models in Electrical Circuits: Theory and Applications*, Amsterdam, the Netherlands: Kluwer Academic Publishers, pp. 321-342.
- Marinov, C. A. and Shivakumar, P. N., Jul. 1996, "Delay time bounds for RC ladder networks," *IEEE Trans. Circuits Systems I: Fundamental Theory and Applications*, vol.43, pp.613-614.
- Mathews, J. H. and Fink, K. D., 1999, *Numerical Methods using MATLAB, Third Edition*, Englewood Cliffs, NJ: Prentice-Hall, Inc., pp. 164-195.

MATLAB User's Guide, 1996, Natick, MA: The Math Works, Inc.

Nakamura, S., 1991, *Applied Numerical Methods with Software*, Englewood Cliffs, NJ: Prentice-Hall, Inc., pp. 312-324.

Oppenheimer, P., 1999, *Top-Down Network Design*, Cisco Press.

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

Raghavan, V., Bracken, J. E. and Rohrer, R. A., Jun. 1992, "AWESpice: A general tool for the accurate and efficient simulation of interconnect problems," *Proceedings of the 29th ACM/IEEE Design Automation Conference*, pp.87-92.

Raghavan, V., Rohrer, R. A., Pillage, L. T., Lee, J. Y., Bracken, J.E. and Alaybeyi, M. M., May. 1993, "AWE-inspired", *Proceedings of IEEE Custom Integrated Circuits Conference*, pp.18.1.1-18.1.8.

Ruston, H. and Bordogna, J., 1966, *Electric Networks: Functions, Filters, Analysis*, Boston, MA: McGraw-Hill Book Company, pp.277-281.

Sakurai, T., Aug. 1983, "Approximation of wiring delay in MOSFET LSI", *IEEE J. Solid-State Circuits*, vol.18, no.4, pp. 418-426.

Schilling, R. J. and Harris, S. L., 2000, *Applied Numerical Methods for Engineers using MATLAB and C*, Pacific Grove, CA: Thomson Publishing, Inc., pp. 224-251.

Schwartz, M., 1987, *Telecommunication Networks Protocol, Modelling and Analysis*, Reading, MA: Addison-Wesley Publishing Company, pp. 70-92.

Shao, J. H. and Chen, R. M. M., May. 2000, "Two-pole approximation for interconnect design and optimization," *International Journal of Electronics*, Oxfordshire, UK: Taylor & Francis, Inc., vol.87, no.5, pp.575-590.

Siebert, W. M., 1994, *Circuits, Signals and Systems*, the MIT Press, New York, NY: McGraw-Hill Book Company, pp.488-490.

Sloman, M., 1994, *Network and Distributed Systems Management*, Reading, MA: Addison-Wesley Publishing Company, pp. 145-165.

References

-
- Spragins, J. D., 2000, *Telecommunications Protocols & Design*, Reading, MA: Addison-Wesley Publishing Company, pp. 76-83.
- Stallings, W., 1999, *SNMP, SNMPv2, SNMPv3 and RMON, 3rd Edition*, Reading, MA: Addison-Wesley Publishing Company, pp. 91-132.
- Sugiuchi, Y., Katz, B. and Rohrer, R. A., Feb. 1994, "Interconnect optimization using asymptotic waveform evaluation (AWE)," *Proceedings of the Multi-Chip Module Conference (IEEE)*, pp.120-125.
- Sungur, M., Ekinici, A. S. and Atalar, A., Jun. 1997, "An efficient program for analysis of interconnect circuits," *International Journal of Electronics*, Oxfordshire, UK: Taylor & Francis, Inc., vol.82, no.6, pp.641-654.
- Tanenbaum, A. S., 2003, *Computer Networks, 4th Edition*, Englewood Cliffs, NJ: Prentice-Hall, Inc., pp. 141-168.
- Varga, R. S., 1962, *Matrix Iterative Analysis*, Englewood Cliffs, NJ: Prentice-Hall.
- Vlach, J. and Singhal, K., 1983, *Computer Methods for Circuit Analysis and Design*, New York, NY: Van Nostrand Reinhold, pp. 252-278.
- Wang, H. J., 1999, *Telecommunications Network Management*, Reading, MA: McGraw-Hill Book Company, pp. 254-289.
- Wang, J. S. H. and Dai, W. W. M., Oct. 1994, "Optimal design of self-damped lossy transmission lines for multichip modules", *Proceedings of the International Conference on Computer Design (IEEE)*, pp.594-598.
- Williams, A. B. and Taylor, F. J., 1995, *Electronic Filter Design Handbook*, New York: NY: McGraw-Hill Book Company, pp.12.47-12.49.
- Xue, T. X., Kuh, E. S. and Yu, Q. J., Feb. 1996, "A sensitivity-based wiresizing approach to interconnect optimization of lossy transmission line topologies," *Proceedings of the Multi-Chip Module Conference (IEEE)*, pp.177-122.
- Yakowitz, S. and Szidarovszky, F., 1989, *An Introduction to Numerical Computations*, Indianapolis, IN: Macmillan Publishing Company, Inc., pp.462.