Neural Network Based Modeling Technique for Modeling Embedded Passives in Multilayer Printed Circuits

by

Xiaolei Ding, B. Eng.,

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirement for the
degree of Master of Applied Science

Ottawa-Carleton Institute for Electrical and Computer Engineering
Department of Electronics
Carleton University
Ottawa, Ontario K1S 5B6
Canada

©Copyright September 2002, Xiaolei Ding

National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisitions et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-79699-X

Canada

The undersigned recommend
the Faculty of Graduate Studies and Research
acceptance of the thesis

**Neural Network Based Modeling Technique for Modeling Embedded
Passives in Multilayer Printed Circuits**

Submitted by

Xiaolei Ding, B. Eng.,

in partial fulfillment of the requirements for
the degree of Master of Applied Science

Q.J. Zhang, Thesis Supervisor

M.C.E. Yagoub, Thesis Co-supervisor

Michel Nakhla, Chair, Department of Electronics

Abstract

Recently, artificial neural networks (ANN) gained attention as a fast and flexible vehicle to microwave circuit design. In this thesis, we present two new approaches to model the electromagnetic effects of embedded passive components in multilayer printed circuit board based on state space equations/equivalent circuit and combined with neural network techniques together. As passives are often connected to nonlinear devices, frequency and time domain models are required. In this paper, an accurate and fast EM based model, developed from S-parameters versus physical/geometrical parameters, is introduced. It can efficiently represent a passive's high frequency behavior in both frequency and time domain circuit design. EM based neural network models, i.e., neural models trained from EM data of microwave components, act as a bridge connecting expensive EM simulation with computer-aided design (CAD) tools. Therefore, Monte-Carlo analysis and yield optimization can be extensively and efficiently used in industry to improve the yield of microwave circuits.

The neural network modeling approach is demonstrated through CAD of an amplifier circuit. Examples of combined model of embedded resistors, capacitors, and the combined models in signal integrity of multilayer circuit are presented to illustrate the advantages of the proposed combined modeling approach.

Acknowledgements

First of all, I would like to express my sincere thanks to my thesis supervisor Professor Qi-Jun Zhang and co-supervisor Professor M.C.E. Yagoub for their professional guidance, continued assistance, invaluable inspiration, motivation, suggestion, and patience throughout the research work and the preparation of this thesis. I am highly indebted to them for having trained me into a full-time researcher with technical, computation and presentation skills, and professionalism.

Next, I would like to thank D. Goulette, R. Sheffield, and H. Feyzbakhsh from Nortel Networks for continued useful discussions, motivation and practical application suggestions.

I am also grateful to Betty Zahalan, Peggy Piccolo, Lorena Duncan, Nagui Mikhail and Jacques Lemieux and all other staff and faculty for providing the excellent lab facilities and friendly environment for study and research. Lei Zhang, Biswarup Chattaraj, Yi Cao, Vijay K. Devabhaktuni and other students in my group are thanked for the help, friendship, and enjoyable collective working.

My special thank is given to Jianjun Xu, without his continued discussion, encouragement, good humor, and delicious Chinese food, working on this thesis would have been very hard.

Finally, I would like to thank my wife Qian Hu, my parents, and my sister, their love, support and encouragement have been and will be the source of strength for overcoming any difficulty and achieving success in my whole life.

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Thesis Objective	3
1.3	Outline of the Thesis	4
2	Literature review	6
2.1	Neural Networks Applications in Microwave/RF Circuit Design	6
2.2	Neural Network Structures	9
2.2.1	Multilayer Perceptrons	9
2.2.2	Knowledge Based Neural Network Model	13
2.2.3	Adjoint Neural Network	16
2.3	Neural Network Training Algorithm	17
2.3.1	Training Objective	17
2.3.2	Review of Back Propagation Algorithm	18
2.3.3	Gradient-based Optimization Methods	19
2.3.4	Global Training Algorithms	23
2.4	Review of Existing Modeling Approaches for Embedded Passives	24

List of Figures

Figure 2.1: Illustration of the feedforward multilayer perceptrons (MLP) structure. Typically, the neural network consists of one input layer, one or more hidden layers, and one output layer	11
Figure 2.2: Illustration of the structure of Knowledge based neural networks (KBNN). The KBNN model includes six layers typically	15
Figure 2.3: The equivalent circuit topology of Caue expansion for an embedded passive model	26
Figure 3.1: Structure of the combined equivalent circuit and neural network model illustrating the model development process and the testing phase. From top to bottom is the combined model training process, which includes parameter extraction and neural network training, and from bottom to top is the combined model feedforward calculation	36
Figure 3.2: The general equivalent circuit of a one port and third order embedded passives example	42
Figure 3.3: Demonstration of the general equivalent circuit structure, which is automatically generated in SSEN model, for a two-port n^{th} order ($n \geq 3$) embedded passives	43

Figure 3.4: Structure of the combined state space equation and neural network model illustrating the model development process and the testing phase. Form top to bottom is the combined model training process, which includes parameter extraction and neural network training. From bottom to top is the combined model feedforward calculation	47
Figure 3.5: EM Based neural network modeling methods for embedded passives	49
Figure 3.6: Flow chart to demonstrate the three methods in modeling embedded passives for time or frequency domain circuit simulation	50
Figure 4.1: 3-D physical structure of embedded resistor. Length and Width are adjustable variables in multilayer circuit design	52
Figure 4.2: The structure of 3-layer MLP Neural model for the embedded resistor. All of the neurons in the structure are fully connected	54
Figure 4.3: The structure of the combined EC-NN model for embedded resistors. The $\mathbf{g}_p$ is a vector that contains all of the lumped components in the user-defined equivalent circuit, as $\mathbf{g}_p = \{R1, R2, C1, C2\}$	56
Figure 4.4: Comparison of real and imaginary of S11 of embedded resistor EC-NN model outputs and independent EM data, which was never used in training. Curves A and B are generated based on $W = 53$ and $L = 11$ mils and $W = 39$ and $L = 9$ mils, respectively	59
Figure 4.5: Comparison of real and imaginary of S21 of embedded resistor EC-NN model outputs and independent EM data, which was never used in training. Curves A and B are generated based on $W = 53$ and $L = 11$ mils and $W = 39$ and $L = 9$ mils, respectively	60

Figure 4.6: Comparison of real and imaginary of S11 of embedded resistor SSE-NN model outputs and independent EM data, which was never used in training. Curves A and B are generated based on $W = 53$ and $L = 11$ mils and $W = 39$ and $L = 9$ mils, respectively	64
Figure 4.7: Comparison of real and imaginary of S21 of embedded resistor SSE-NN model outputs and independent EM data, which was never used in training. Curves A and B are generated based on $W = 53$ and $L = 11$ mils and $W = 39$ and $L = 9$ mils, respectively	65
Figure 4.8: The 3 rd order SSE-NN model in frequency-domain simulation and y_{jk} ($j, k = 1, 2$) are real part of the Y matrix elements. The W is sweep from 35 to 55 mils. Let define $D = y_{11}y_{22} - y_{12}y_{21}$, the model is passive because D is always greater than zero	66
Figure 4.9: 3-D physical structure of embedded capacitor. Length (L) and capacitor dielectric constant (ϵ_{rcap}) are adjustable variables in multilayer circuit design	67
Figure 4.10: 3-layers MLP Neural model of the capacitor	69
Figure 4.11: The combined EC-NN model structure for embedded capacitor. The equivalent circuit is user defined. Let $\mathbf{g}_p = \{C1, C2, L1\}$, a lumped component values vector is calculated by MLP in the combined EC-NN model	71
Figure 4.12: Comparison of S11 of embedded resistor EC-NN model outputs and independent EM data, which was never used in training. Curves A and B are generated based on $L = 29$ mil, $\epsilon_{\text{rcap}} = 15$ and $L = 31$ mil, $\epsilon_{\text{rcap}} = 20$ respectively	72

Figure 4.13: Comparison of S21 of embedded capacitor EC-NN model outputs and independent EM data, which was never used in training. Curves A and B are generated based on $L = 29\text{mil}$, $\epsilon_{\text{rcap}} = 15$ and $L = 31\text{mil}$, $\epsilon_{\text{rcap}} = 20$ respectively	73
Figure 4.14: Comparison of S11 of embedded capacitor SSE-NN model outputs and independent EM data, which was never used in training. Curves A and B are generated based on $L = 29\text{mil}$, $\epsilon_{\text{rcap}} = 15$ and $L = 31\text{mil}$, $\epsilon_{\text{rcap}} = 20$ respectively	75
Figure 4.15: Comparison of S21 of embedded capacitor SSE-NN model outputs and independent EM data, which was never used in training. Curves A and B are generated based on $L = 29\text{mil}$, $\epsilon_{\text{rcap}} = 15$ and $L = 31\text{mil}$, $\epsilon_{\text{rcap}} = 20$ respectively	76
Figure 4.16: The amplifier circuit, in which, all the passive components are represented by the pure neural network models. N_R and N_C denote neural models of embedded resistors and capacitors respectively	77
Figure 4.17: 3-D illustration of physical connections between a few selected components in the amplifier circuit	78
Figure 4.18: Gain vs. frequency for <i>ADS</i> original circuit (—○—) and Monte-Carlo analysis using neural models (—)	81
Figure 4.19: Three dimensional illustration of signal integrity of multilayer circuit with embedded resistor and capacitor	83
Figure 4.20: Comparison of signal from input buffer, and output signals before and after combined SSE-NN models optimization	83

Figure 4.21: The three coupled transmission line circuit	85
Figure 4.22: 500 output curves vs. time for <i>Hspice</i> Monte-Carlo analysis of the 3-coupled transmission line example using our combined SSE-NN models of embedded passives	86

List of Tables

Table 4-I: MLP training data, which is collected from parameter extraction, includes geometrical parameter and optimized lumped component values in the user defined equivalent circuit through the entire frequency range	58
Table 4-II: Comparison of resistor combined model with different order formulation	62
Table 4-III: The training data, which generated from parameter extraction based on 3 rd order state space formulation, is used to train a MLP neural network in SSE-NN model for the embedded resistor. Each set of coefficient values are optimized based on each geometry of embedded resistor within the entire frequency range	63
Table 4-IV: The three LHP poles of the 3 rd order SSE-NN model	66
Table 4-V: Comparison of capacitor combined model with different order formulation	74
Table 4-VI: The geometry of embedded passives neural model before and after nominal optimization	79
Table 4-VII: The comparison of geometry of embedded passives before and after yield optimization in the amplifier circuit	80
Table 4-VIII: Comparison of the physical/geometrical input parameters in embedded passives before optimization and after optimization	84

Chapter 1

Introduction

1.1 Motivations

The effective use of Computer Aided Design (CAD) tools in both electrical and physical design stages is important in designing RF and Microwave circuits and systems with shrinking design margins and expanding system complexities. The need of reducing design iterations of such system furthers demands that the tools be fast and reliable. As the signal speed and frequency increase, the dimensions of interconnects and passive components in multilayer circuits become a significant fraction of signal wavelength. The conventional electrical models for the components are not accurate anymore. Models with physical/geometrical information, and including electromagnetic (EM) effects, become necessary. However, detailed EM and physical-based models are computationally intensive because the model evaluation typically involves numerically solving partial differential equations. On the other hand, the desire to reduce time-to-market leads to the trend of increased use of upfront computer simulation instead of hardware prototyping. Many behaviors of the circuits and systems, such as performance, manufacturability, and electrical and physical reliabilities, can be predicted by computers before hardware implementation. Statistical analysis and yield optimization are two key

components in modern CAD tools in order to meet these requirements. Statistical design is a highly repetitive process in which component models and circuits need to be solved repetitively following known or assumed statistical distributions leading to more intensive computations.

Embedded passives represent an emerging technology area that has the potential for increased reliability, improved electrical performance, shrunk size, and reduced cost. The conventional approach for circuit and system design requires equivalent circuit to capture the response of embedded passives. In conventional simulators, passive components can be modeled as ideal lumped components or equivalent circuit models. In high frequency, the EM effects in embedded resistors and capacitors become significant. Therefore, EM representation of circuit components becomes important. However, the existing equivalent circuit method may not be accurate enough to reflect high frequency EM effects. Even if we can find an accurate equivalent circuit to represent high frequency EM effects, the component values in the equivalent circuit do not directly represent the embedded passives' structural geometrical/physical parameters. If an embedded passive is used in circuit statistical design, we have to generate a set of lumped component values of the given equivalent circuit to match a new geometry of the embedded passive when the geometrical parameters are changed.

Fundamental of CAD tools is that all components are represented by a set of mathematical equations. The accuracy and efficiency of model evaluations play an important role in many levels of circuit design including modeling, simulation, nominal optimization, statistical design, and yield optimization. Therefore, trying to find an

approach that can efficiently develop fast and accurate model for embedded passives in microwave circuit design is a primary motivation of this thesis.

Artificial neural networks (ANN) are information processing systems inspired by the ability of human brain to learn from observation and to generalize by abstraction. The fact that neural networks can be trained for totally different applications led to their use in diverse fields such as pattern recognition, speech processing, control, medical application and more. Since neural networks were introduced into RF/microwave area in 1993, they have gradually gained attention as a fast and flexible vehicle in circuit modeling, simulation, and optimization. Neural networks can learn and generalize from data that are obtained through solving original EM and physical problems or by measurement. Neural networks can be used for rapid estimation of microwave component and circuit performance in highly repetitive design process such as optimization and statistic analysis. How to fully explore the advantages of neural networks and exploit the existing techniques to develop reliable embedded passive models that can be efficiently used in both time- and frequency-domain CAD tools is another motivation in this thesis.

1.2 Thesis Objective

The main objective of this thesis is to use artificial neural network techniques to develop embedded passive models for RF/Microwave circuit design in time and frequency domain circuit simulators. In this thesis, the following substantial work of neural network based modeling approaches for embedded passives are presented:

- (1) : Use Multilayer Perceptrons (MLP) structure to train ANN models of embedded resistors and capacitors for printed circuit board (PCB) design in frequency domain. The advantages of MLP are demonstrated through optimization, Monte-Carlo analysis, and yield optimization in frequency domain CAD tools such as ADS [1].
- (2) : A novel hierarchical modeling approach, called combined model, which combines state space equation and MLP to represent embedded passive EM behavior in time and frequency domain circuit design is proposed. A detailed formulation of this novel hierarchical structure, featuring state space equation to automatically generate a general equivalent circuit for the embedded passive, is introduced [2].
- (3) : Another combined neural model structure, which combines empirical equivalent circuit and neural network together, is developed for efficiently modeling embedded passives in both time and frequency domain circuit design.
- (4) : A training method for combined model structure featuring gradient based optimization is used. In the combined model training process, a set of constraints are used to guarantee the combined model is stable and passive [2].

1.3 Outline of the Thesis

The thesis is organized as follows.

In chapter 2, a detailed review of different neural network structures and training algorithms is presented. A review of existing modeling approaches for embedded passives of RF/microwave circuit design is also conducted.

Chapter 3 introduces three ANN based modeling techniques. We use MLP to develop embedded passive models in frequency domain. In addition, a novel hierarchical combined model structure for time/frequency domain circuit simulators is presented. The structure features state space theory or equivalent circuit and artificial neural networks. A detailed description and formulation of each type of combined structures are presented. A general equivalent circuit topology for any embedded passive is introduced. Stability and passivity are satisfied for time domain simulation.

In chapter 4, ANN based models for embedded passives, which include pure neural network models and combined state space theory/equivalent circuit and neural network models, are introduced. The advantages of ANN based modeling approach are demonstrated through embedded resistor and capacitor models. Accuracy comparison is presented in this chapter. Amplifier and signal integrity analysis in multilayer printed circuit examples are used at the end of the chapter to illustrate advantages of the proposed modeling approach.

Finally, in chapter 5, conclusions and suggestions for future research are discussed.

Chapter 2

Literature Review

As neural network techniques are relatively new to the microwave community, it is not easy for microwave designer to make decisions regarding these issues. For example, the structure of neural network, the activation function in the hidden neurons, the training process, training algorithms, etc., are not obvious. All of these issues are briefly discussed in the following section. In this chapter, we also briefly review some existing modeling methods that represent EM effects of embedded passives in microwave circuit design.

2.1 Neural Network Applications in Microwave/RF Circuit

Design

The drive in the microwave industry to meet the demands of high manufacturability and fast design cycles created a need for efficient statistical design techniques. Statistical analysis and yield optimization that take into account the manufacturing tolerance, model uncertainties, variation in the process parameters, etc., are widely accepted as indispensable components of the circuit design methodology [1] - [6]. Detailed physical EM models of active/passive components can be an important step towards a design for the first-pass success, but the models are computationally intensive. Recently, neural

network technology has been introduced in the microwave modeling, simulation, and optimization. Neural networks possess the distinguished ability of learning from samples of input-output data to accurately model nonlinear relationships. Neural models are much faster than original detailed physical/EM models [5] [7], and more accurate than polynomial and empirical models [8], allow more dimensions than table lookup models [9], and are easier to develop when a new device/technology is introduced [10]. Once developed, these neural network models can be used in place of computationally intensive physical/EM models of active and passive components [5] [7] [11] to speed up microwave circuit design. Recent work by microwave researchers demonstrated the ability of neural networks to accurately model a variety of microwave components, such as microstrip interconnects [7], vias [12], spiral inductors [11], FET devices [7] [13], HBT devices [14], HEMT devices [15], filters [16], amplifiers [5] [17], coplanar waveguide (CPW) circuit components [18], mixers [17], antennas [19], embedded resistors [1] - [3], packaging and interconnects [20], etc. Neural networks have also been used in circuit simulation and optimization [5] [21], signal integrity analysis and optimization of VLSI interconnects [20] [22], microstrip circuit design [23], process design [24], synthesis [25], and microwave impedance matching [26]. These pioneering works have established the framework of neural modeling technique in both device and circuit level of microwave applications.

In the development of neural network model for solving microwave problems, the two most important factors are neural network structures and training algorithms. In order to represent the device/circuit behavior, the neural network has to be trained with the corresponding data. A good collection of the training data, which is well distributed,

sufficient, and accurately measured/simulated, is the basic requirement to obtain an accurate model. However, training data collection/generation may be very expensive in the reality of microwave problem. There is a tradeoff between the amount of training data needed for developing the neural model and the accuracy required by the application. Because different neural network structures have different efficiency to represent microwave problem, an appropriate structure would help to achieve higher model accuracy with fewer training data [27]. For example, a feedforward neural network with smooth switching functions in the hidden layer is good for modeling smooth, slowly varying nonlinear functions, while a feedforward neural network with Gaussian functions in the hidden layer could be more effective in modeling nonlinear functions with large variations. Furthermore, for knowledge based neural network, less training data is required as well as good accuracy can be achieved [7]. The size of the structure, i.e., the number of hidden neurons, and the number of hidden layers, are also not easy to determine or predict in the development of a neural network model. The problems of underlearning and overlearning have to be dealt with, i.e., small size neural network cannot learn the problem very well (under-learning), and however, too complicated structure will lead to over-learning.

Training is an essential step in neural network model development. An appropriate structure cannot guarantee to achieve a good model, unless trained by a suitable training algorithm. A suitable training algorithm can speed up the training procedure with better accuracy. In the middle of 1980's, one of the most popular neural networks training algorithms, Back Propagation (BP), was proposed. Later, a lot of variations to improve the convergence of BP were proposed. Optimization methods such as second order

methods and decomposed optimization have also been used for neural network training in recent years. Recently, global optimization techniques such as Genetic algorithm and Simulated Annealing have been combined with conventional training algorithms to solve the problem of numerous local minimal in neural network training procedure.

2.2 Neural Network Structures

A variety of neural network structures have been developed in the neural network community for microwave circuit, signal processing, control, etc. Several neural network structures are described in this section, which include multilayer perceptrons (MLP), knowledge-based neural networks (KBNN), and adjoint neural networks.

2.2.1 Multilayer Perceptrons (MLP)

An important class of feedforward neural networks, which is a basic type of neural networks capable of approximating generic continuous and integrable functions, is multilayer perceptrons. Typically, the MLP neural network consists of an input layer, one or more hidden layers and an output layer, as shown in Fig. 2-1. Suppose the total number of layers is L . The input layer is layer 1, the output layer is layer L and hidden layers are 2, 3, ..., $L-1$. Let the number of neurons in the l^{th} layer be N_l , $l = 1, 2, \dots, N_h$. Let w_{ij}^l represent the weight of the link between j^{th} neuron of $(l-1)^{th}$ hidden layer and i^{th} of l^{th} hidden layer, and θ_i^l be the bias parameter of i^{th} neuron of l^{th} hidden layer. Let x_i represents the i^{th} input parameter to

the MLP. Let $\hat{y}_i^l$ be the output of i^{th} neuron of l^{th} hidden layer, which can be computed according to the standard MLP formulae as

$$\hat{y}_i^l = \sigma\left(\sum_{j=1}^{N_{l-1}} w_{ij}^l \cdot \hat{y}_j^{l-1} + \theta_i^l\right), \quad i=1,2,\dots,N_L, \quad l=1,2,\dots,N_h \quad (2.1)$$

$$\hat{y}_i^1 = x_i, \quad i=1,2,\dots,N_I \quad (2.2)$$

where $\sigma(\cdot)$ is the activation function of hidden neurons and N_I is the number of MLP model's inputs. Let v_{ki} represents the weight of the link between i^{th} neuron of N_h^{th} hidden layer and k^{th} neuron of output layer, and β_k be the bias parameter of k^{th} output neuron. The outputs of MLP can be computed as

$$\hat{y}_k = \sum_{i=1}^{N_{N_h}} v_{ki} \cdot \hat{y}_i^{N_h} + \beta_k, \quad k=1,2,\dots,N_y. \quad (2.3)$$

For function approximation, output neurons can be processed by linear function as shown in (2.3). The most commonly used activation function $\sigma(\cdot)$ for hidden neurons is the logistic sigmoid function given by

$$\sigma(\gamma) = \frac{1}{(1 + e^{-\gamma})} \quad (2.4)$$

which has property

$$\sigma(\gamma) \rightarrow \begin{cases} 1 & \text{as } \gamma \rightarrow +\infty \\ 0 & \text{as } \gamma \rightarrow -\infty \end{cases} \quad (2.5)$$

Other possible candidates for $\sigma(\cdot)$ are the arctangent function

$$\sigma(\gamma) = \left(\frac{2}{\pi}\right) \arctan(\gamma) \quad (2.6)$$

and the hyperbolic tangent function

Fig. 2-1 : Illustration of the feedforward multilayer perceptrons (MLP) structure. Typically, the neural network consists of one input layer, one or more hidden layers, and one output layer.

$$\sigma(\gamma) = \frac{(e^\gamma - e^{-\gamma})}{(e^\gamma + e^{-\gamma})} \quad (2.7)$$

All these functions are bounded, continuous, monotonic and continuously differentiable.

The universal approximation theorem states that there always exists a 3-layer MLP that will approximate to any accuracy desired by the user for any arbitrary nonlinear, continuous, multi-dimensional function [28] [29]. However, the universal approximation theorem does not tell us the number of neurons needed. Therefore, failure to develop an accurate neural model can be attributed to inadequate learning, unsatisfactory number of hidden neurons, or the presence of stochastic rather than a deterministic relationship between inputs and outputs [30]. In practice, the number of hidden neurons depends on the degree of nonlinearities and the dimensions of the original problem. But too many hidden neurons will lead to overlearning in the training process.

Neural networks with one or two hidden layers, i.e., 3-layer or 4-layer MLP are more frequently used and are usually suitable for RF/microwave application. The performance of neural network can be evaluated in terms of generalization capability and mapping capability. In [31], Tamura demonstrated that 3-layer MLP is preferred in function approximation where generalization capability is a major concern. Intuitively, 4-layer MLP would perform better in nonlinear problems in which localized behavioral components exist repeatedly in different regions of the problem space.

2.2.2 Knowledge-Based Neural Network Models (KBNN)

MLP is a kind of black-box model structurally embedding no-problem dependant information. A large amount of training data is usually needed to ensure model accuracy. However, generating large amounts of training data could be very expensive for microwave problems, e.g., EM simulation could be very expensive to generate many points in the model input parameter space. Existing microwave knowledge can provide additional information of the original problem that may not be adequately represented by the limited training data. In KBNN, the neural network can help bridge the gap between empirical model and EM solutions.

The structure of knowledge-based neural networks (KBNN) [7] is illustrated in Fig. 2-2. The microwave knowledge is embedded as a part of the overall neural network internal section. There are six layers, which are not fully connected to each other, in the KBNN structure, namely input layer X , knowledge layer Z , boundary layer B , region layer R , normalized region layer R' and output layer $\hat{Y}$. The knowledge layer Z is the place where microwave knowledge resides in the form of single or multidimensional function $\psi(\cdot)$. For knowledge neuron i in the Z layer

$$z_i = \psi_i(\mathbf{x}, \mathbf{w}_i), \quad i = 1, 2, \dots, N_z \quad (2.8)$$

where $\mathbf{x}$ is a neural network inputs vector, N_z is the number of knowledge neurons, and $\mathbf{w}_i$ is a vector of parameters in the knowledge formula. The knowledge function $\psi_i(\mathbf{x}, \mathbf{w}_i)$ is usually in the form of empirical or semi-analytical functions. The boundary layer B can incorporate knowledge in the form of problem dependent boundary functions $B(\cdot)$. Neuron i in the layer B is calculated by

$$b_i = B_i(\mathbf{x}, \mathbf{v}_i), \quad i = 1, 2, \dots, N_b \quad (2.9)$$

where v_i is a vector of the parameters in B_i defining an open or closed boundary in the input space x . The region layer R contains neurons to construct regions from boundary neurons,

$$r_i = \prod_{j=1}^{N_b} \sigma(\alpha_{ij} b_j + \theta_{ij}), \quad i = 1, 2, \dots, N_r \quad (2.10)$$

where α_{ij} and θ_{ij} are the scaling and bias parameters, respectively. The normalized region layer R' contains rational function-based neurons to normalize the outputs of region layer,

$$r'_i = \frac{r_i}{\sum_{j=1}^{N_r} r_j}, \quad i = 1, 2, \dots, N_r \quad \text{where } N_r = N_r \quad (2.11)$$

The output layer $\hat{y}$ contains second-order neuron combining knowledge neurons and normalized region neurons

$$\hat{y}_j = \sum_{i=1}^{N_t} \beta_{ji} z_i \left(\sum_{k=1}^{N_r} \rho_{jik} r'_k \right) + \beta_{j0}, \quad j = 1, 2, \dots, N_y \quad (2.12)$$

where β_{ji} reflects the contribution of the i^{th} knowledge neurons to output neuron $\hat{y}_j$ and β_{j0} is the bias parameter, ρ_{jik} is one indicating that region r'_k is the effective region of the i^{th} knowledge neuron contributing to the j^{th} output. A total of N_r regions are shared by all the output neurons. As a special case, if we assume that each normalized region neuron selects a unique knowledge neuron for each output j , the function output neurons can be simplified as

$$\hat{y}_k = \sum_{i=1}^{N_t} \beta_{ki} z_i r'_i + \beta_{k0}, \quad k = 1, 2, \dots, m \quad (2.13)$$

Compare to pure neural network structures, the prior knowledge in KBNN gives neural network more information about the original microwave problem, besides the information

Fig. 2-2 : Illustration of the structure of Knowledge based neural networks (KBNN). The KBNN model includes six layers typically.

included in the training data. Consequently, KBNN models have better reliability when training data is limited or when the model is used beyond training range.

2.2.3 Adjoint Neural Networks

Adjoint neural network can provide sensitivity information for circuit optimization and modeling [32]. Two neural networks work together to calculate sensitivity analysis, one called original neural network and another called adjoint neural network. The original neural network can be any kind of neural network structure such as MLP, radial-basis function network, KBNN, etc. Using second order derivative information, the neural network can be trained to learn not only device input/output relationship but also the derivative information, which is very useful in simultaneous DC/small-signal/large-signal device modeling.

Let $\mathbf{d}$ and $\mathbf{d}'$ represent the training data for the original output $\hat{\mathbf{y}}$ and its derivatives $d\hat{\mathbf{y}}/d\mathbf{x}$, respectively. Let I , K and S be the index sets of input and output neurons, and samples in training data $\mathbf{d}$, respectively. We formulate the error function for training as,

$$E = \frac{1}{2} \sum_{s \in S} \left[p_1 \sum_{k \in K} (\hat{y}_{ks} - d_{ks})^2 + p_2 \sum_{i \in I, k \in K} \left(\frac{d\hat{y}_{ks}}{dx_{is}} - (d')_{kis} \right)^2 \right] \quad (2.14)$$

where subscripts i , k and s (used for x , $\hat{\mathbf{y}}$, $\mathbf{d}$ and $\mathbf{d}'$) indicate input neuron i , output neuron k and sample s , respectively, and p_1 , p_2 are the weighting parameters. During training, both the original and the adjoint neural models share the same set of parameters w_i , $i = 1, 2, \dots, N$. Therefore training one model will also result in the other model being updated. There are three types of training methods. (i) Train original neural model using input/output data $\mathbf{d}$, and after training, the outputs of adjoint model automatically become derivatives of original input/output. (ii) Train adjoint model only with derivative data

$d\hat{y}/dx$. The original model will then give original input/output (i.e., x - $\hat{y}$) relationship, which has the effect of providing integration solution over derivative training data. (iii) Train both original and adjoint models together to learn x - $\hat{y}$ and $d\hat{y}/dx$ data, which will help the neural model to be trained more accurately and robustly.

2.3 Neural Network Training Algorithm

2.3.1 Training Objective

A neural network model can be developed through an optimization process called training. Let us consider the training data including N_p sample pairs, $\{(\mathbf{x}_p, \mathbf{d}_p), p = 1, 2, \dots, N_p\}$, where $\mathbf{x}_p$ and $\mathbf{d}_p$ are N_x and N_y dimensional vectors representing the inputs and outputs of the neural network respectively. Let $\mathbf{w}$ represent the parameters inside the neural network called as the weight vector. Let $\hat{\mathbf{y}} = f_{ANN}(\mathbf{x}, \mathbf{w})$ represent the input-output relationship of the neural network.

The objective of training is to find $\mathbf{w}$ such that the error between neural network predictions and the desired output are minimized

$$\min_{\mathbf{w}} E(\mathbf{w}) = \sum_{i=1}^{N_y} e_i(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^{N_y} (\hat{y}_{pk} - d_{pk})^2 \quad (2.15)$$

where d_{pk} is the k^{th} element of vector $\mathbf{d}_p$, $\hat{y}_{pk}$ is the k^{th} output of the neural network when the input is $\mathbf{x}_p$. The objective of neural network training is to adjust neural network connection weights $\mathbf{w}$ such that $E(\mathbf{w})$ is minimized.

The objective function $E(\mathbf{w})$ is a nonlinear function w.r.t. the adjustable weight vector $\mathbf{w}$. Due to the complexity of $E(\mathbf{w})$, iterative algorithms are often used to explore the parameter space efficiently.

2.3.2 Review of Back Propagation Algorithm

In 1986, Rumelhart, Hinton and Williams proposed a steepest descent algorithm for neural network training called Back Propagation (BP) [33]. In each step of the step-by-step algorithms, the BP is first done layer by layer to calculate the derivation of cost function $E(\mathbf{w})$ to weights $\mathbf{w}$. The weights of the neural network $\mathbf{w}$ are updated along the negative gradient direction in the weight space. The update formula are given by

$$\Delta \mathbf{w}_{now} = \mathbf{w}_{next} - \mathbf{w}_{now} = -\eta \frac{\partial e_i(\mathbf{w})}{\partial \mathbf{w}} \Big|_{\mathbf{w}=\mathbf{w}_{now}} \quad (2.16)$$

$$\Delta \mathbf{w}_{now} = \mathbf{w}_{next} - \mathbf{w}_{now} = -\eta \frac{\partial E(\mathbf{w})}{\partial \mathbf{w}} \Big|_{\mathbf{w}=\mathbf{w}_{now}} \quad (2.17)$$

where constant η , called learning rate, controls the step size of weight update. In formula (2.16), the weights are updated after each sample has been used to teach the neural network. Update formula (2.17) is called batch mode update, where the weights are updated after all training samples have been presented to the network.

The learning rate η is a sensitive parameter in the back propagation algorithm. If η is a small number, more iteration is needed for training. However, if η is too large to speed up the learning process, training may be unstable due to weight oscillation [34]. The addition of a momentum term to weight update formula in (2.16) and (2.17), as proposed by [33], provides significant improvement to the BP, reducing the weight oscillation.

$$\Delta \mathbf{w}_{now} = -\eta \frac{\partial e_i(\mathbf{w})}{\partial \mathbf{w}} \Big|_{\mathbf{w}=\mathbf{w}_{now}} + \alpha (\mathbf{w}_{now} - \mathbf{w}_{old}) \quad (2.18)$$

$$\Delta \mathbf{w}_{now} = -\eta \frac{\partial E(\mathbf{w})}{\partial \mathbf{w}} \Big|_{\mathbf{w}=\mathbf{w}_{now}} + \alpha (\mathbf{w}_{now} - \mathbf{w}_{old}) \quad (2.19)$$

where the constant α is the momentum factor which controls the influence of the last weight update direction on the current weight update, w_{old} represents the last point of w . To reduce the weight oscillation, many approaches have been proposed, such as invoking a correction term that uses the difference of gradients [35], and placing an extra constraint so that the alignments of successive weight updates are maximized [36].

An important way to efficiently train a neural network by BP is to use adaptation schemes that allow the learning rate and the momentum factor to be adaptive during learning [37], e.g., adaptation according to training errors [38]. The adaptation is determined by two factors, one being the current derivative of the training error with respect to the weights, and the other being an exponentially weighted sum of the current and past derivatives of the training error. In [39], which is considered as an extension of Jacob's heuristics, the training scheme corrects the values of weights near the bottom of the error surface ravine with a new acceleration algorithm. This correction term uses the difference between gradients to reduce the weight oscillation during training.

2.3.3 Gradient-based Optimization Methods

The Back Propagation algorithm, which is based on steepest descent principle, is relatively easy to implement. However, the error surface of neural network training usually contains planes with a gentle slope due to the squashing functions commonly used in neural networks. The error gradient values are too small for weight to move rapidly on these planes, the rate of convergence is slow. The rate of convergence could be very slow when the steepest decent method encounters "narrow valley" in the error

surface where the direction of gradient is close to the perpendicular direction of the valley. The update direction oscillates back and forth along the local gradient.

Because supervised learning of neural networks can be considered as a function of optimization problem, higher order optimization methods using gradient information can be adopted in neural network training to improve the rate of convergence. Compared to the heuristic approach discussed in the earlier Back Propagation section, these methods have a sound theoretical basis and guaranteed convergence for most of the smooth functions. Some of the early work in this area was demonstrated in [40] [41] with the development of second-order learning algorithms for neural networks. In [42], various first- and second-order optimization methods for feedforward neural network training are reviewed.

As mentioned before, the cost function $E(\mathbf{w})$ of training is defined as the summation of the square of difference between output of neural networks and training data. It is a parametric function depending on all the training parameters in the neural network. The purpose of training is to find a minimum point $\mathbf{w} = \mathbf{w}^*$ in the parameter space that minimizes $E(\mathbf{w})$. Let $\mathbf{h}$ be the direction vector, η be the learning rate, $\mathbf{w}_{now}$ be the current value of $\mathbf{w}$, then the optimization will update $\mathbf{w}$ such that

$$E(\mathbf{w}_{next}) = E(\mathbf{w}_{now} + \eta \mathbf{h}) . \quad (2.20)$$

The principal difference between various descent algorithms lies in the procedure to determine successive update directions ($\mathbf{h}$) [43]. Once the update direction is determined, the optimal step size could be found by line search along $\mathbf{h}$,

$$\eta^* = \min_{\eta > 0} E(\eta) \quad (2.21)$$

where

$$E(\eta) = E(\mathbf{w}_{now} + \eta \mathbf{h}) \quad (2.22)$$

When downhill direction $\mathbf{h}$ is determined from the gradient $\mathbf{g}$ of the objective function E , such descent methods are called as gradient-based descent methods. The procedure for finding a gradient vector in a network structure is generally similar to Back Propagation in the sense that the gradient vector is calculated in the direction opposite to the flow of output from each neuron.

A. Conjugate Gradient Training Algorithm

The conjugate gradient method is originally derived from quadratic minimization and the minimum of the objective function E can be efficiently found within N_w iterations. With

initial gradient $\mathbf{g}_{initial} = \frac{\partial E}{\partial \mathbf{w}} \Big|_{\mathbf{w}=\mathbf{w}_{initial}}$, and direction vector $\mathbf{h}_{initial} = -\mathbf{g}_{initial}$, the conjugate

gradient method recursively constructs two vectors sequence [44],

$$\mathbf{g}_{next} = \mathbf{g}_{now} + \lambda_{now} \mathbf{H} \mathbf{h}_{now} \quad (2.23)$$

$$\mathbf{h}_{next} = -\mathbf{g}_{now} + \gamma_{now} \mathbf{h}_{now} \quad (2.24)$$

$$\lambda_{now} = \frac{\mathbf{g}_{now}^T \mathbf{g}_{now}}{\mathbf{h}_{now}^T \mathbf{H} \mathbf{h}_{now}} \quad (2.25)$$

$$\gamma_{now} = \frac{\mathbf{g}_{next}^T \mathbf{g}_{next}}{\mathbf{g}_{now}^T \mathbf{g}_{now}} \quad (2.26)$$

or,

$$\gamma_{now} = \frac{(\mathbf{g}_{next} - \mathbf{g}_{now})^T \mathbf{g}_{next}}{\mathbf{g}_{now}^T \mathbf{g}_{now}} \quad (2.27)$$

where $\mathbf{h}$ is called the conjugate direction and $\mathbf{H}$ is the Hessian matrix of the objective function E . Here, (2.26) is called the Fletcher-Reeves formula and (2.27) is called the

Polak-Ribiere formula. To avoid the need of hessian matrix to compute the conjugate direction, we proceed from w_{now} along the direction h_{now} to the local minimum of E at w_{next} through line minimization, and then set $g_{next} = \frac{\partial E}{\partial w} \Big|_{w=w_{next}}$. This g_{next} can be used as the vector of (2.23), and as such (2.25) is no longer needed. We can make use of this line minimization concept to find conjugate direction in neural network training, thus avoiding intensive Hessian matrix computations. In this method, the descent direction is along the conjugate direction that can be accumulated without computations involving matrices. As such, conjugate gradient methods are very efficient and scale well with the neural network size.

B. Quasi-Newton Training Algorithm

Quasi-Newton algorithm is also derived from quadratic objective function optimization. The inverse of Hessian matrix $A = H^{-1}$ is used to bias the gradient direction. In Quasi-Newton training method, the weights are updated by

$$w_{next} = w_{now} - \eta A_{now} g_{now} \quad (2.28)$$

Standard quasi-Newton methods require N_w^2 storage space to maintain an approximation of the inverse Hessian matrix, where N_w is the total number of weights in the neural network structure, and a line search is indispensable to calculate a reasonably accurate step length. A reasonable accurate step size is efficiently calculated in one-dimensional line search by a second-order approximation of the objective function. Through the estimation of inverse Hessian matrix, Quasi-Newton has faster convergence rate than conjugate gradient method.

2.3.4 Global Training Algorithm

Another important class of methods uses random optimization techniques that are characterized by a random search element in the training process allowing the algorithms to escape from local minimal and converge to the global minimum of the objective function. Two representative algorithms in this class are the simulated annealing algorithm and the genetic algorithm. In [45], the simulated annealing algorithm that analogs the annealing process of atoms inside metal is presented. The random parameter in the optimization process is controlled by a parameter, which determines the search range that a local minimum can jump out. In [46], the genetic training algorithm, which evolves the structure and weights of neural network through generations in a manner that is similar to biology evolution, is introduced to train the neural network.

Because of the random search during the neural network training, the convergence of these training methods is slow. A hybrid method that combines the conjugate gradient method and random optimization is proposed in [47]. During the training with conjugate gradient method, if a flat error surface is encountered, the training algorithm switches to the random optimization method. After the training escapes from the flat error surface, it switches back to the conjugate gradient method.

We have discussed many issues about the neural network in structure, activation function, and training algorithm. In next step, we will present some existing modeling methods for embedded passives in RF/microwave design.

2.4 Review of Existing Modeling Approaches for Embedded Passives

Numerical electromagnetic methods are usually computationally time consuming for embedded passives [48]. In addition, EM effects generally need to be completely re-simulated even if the geometrical/physical input parameters are changed slightly. Furthermore, most full wave solvers generate field patterns or S-parameters that may not be easy to incorporate in a standard circuit simulator, such as SPICE that are widely used for circuit design applications. Therefore, lumped equivalent circuit models for embedded passives are common methods for capturing high frequency effects of passives in microwave and RF circuit design. Many methods of extracting equivalent circuit have been studied such as network synthesis, optimization methods, and unit block methods.

2.4.1 Equivalent Circuit Generated from Network Synthesis

Many lumped equivalent circuits of embedded passives are generated by synthesizing rational function. S-parameters are generated by full-wave frequency domain EM simulations or measurements. In [49], to represent the acquired information in a physically meaningful format, the S-parameters need to be converted to the impedance parameters. Then, the impedance parameter Z_{ij} can be characterized by the following partial fraction expansion

$$Z_{ij}(s) = \frac{k_{ij}^{(0)}}{s} + \sum_{m=1}^M \frac{2k_{ij}^{(m)}s}{s^2 + \omega_m^2} + \dots + k_{ij}^{(\infty)}s \quad (2.29)$$

where $s = j\omega$, $k_{ij}^{(m)}$ is residue of poles and M is the number of contribution from the intermediate pole of the complex frequency ω_m .

Once the expansions have been obtained for each impedance parameter, the residues and poles can be used to determine the equivalent circuit component values. For each term in the Cauer expansion, there will be a corresponding sub-network. For a one pole linear circuit, the equivalent circuit contains three sub networks, one for the residue at zero frequency ω_0 , one for the residue at infinite frequency ω_{inf} , and one for the intermediate pole occurring at the frequency ω_l . A two-port M poles linear circuit is shown in Fig. 2-3, each block in sub network is corresponding to zero, infinite, and M intermediate poles in the band of interest.

This form may be expanded to an arbitrary number of ports. For each branch (block), the two-port matrix may be expressed as

$$Z_{11} = Z_a + Z_c \quad (2.30)$$

$$Z_{12} = aZ_c = Z_{21} \quad (2.31)$$

$$Z_{22} = a^2 Z_b + a^2 Z_c \quad (2.32)$$

where Z_a , Z_b and Z_c represent either an inductor, capacitor, resistor, or a LC combination. For the process of equivalent circuit extraction, we first determine a set of branch components that produce the same response obtained from full-wave EM simulation. Once the Cauer expansion has been obtained, we can use the residue values of each pole function of Z_{11} , Z_{12} and Z_{22} to calculate the equivalent lumped element circuit parameters.

Another network synthesis method is introduced in [50]. K.L. Choi generated a rational function to present EM behavior. Since a transient waveform has real values, the rational polynomial has to be suitably modified to include complex conjugate poles and zeros. Their technique contribution is that using minimum EM data to produce an

Fig. 2-3: The equivalent circuit topology of Cauer expansion for an embedded passive model.

accurate polynomial rational function to represent the embedded components in the desired frequency range. Based on their experience, no more than four poles were required to develop a model and achieve an acceptable response in the range of DC - 20 GHz. The lumped elements based on the rational function are then extracted for the equivalent circuit.

In [51], a method is presented to use rational function, which is generated from EM data, and then to synthesize equivalent circuit. The method does not guarantee a stable solution, especially when high order approximation is desired. Therefore, a stability enforcement algorithm is applied to the unstable function to make the model stable. The unstable poles of the rational function are discarded and only stable poles are collected.

2.4.2 Optimization Based Equivalent Circuit Generation

Once we define the equivalent circuit model for an embedded passive, we need to calculate S-parameters and their derivatives to circuit parameters. The derivative will be used in the optimization procedure for searching deep-step direction [52]. The S-parameters and derivatives can be defined as

$$S_{11} = \frac{2V_1}{Z_0 I} - 1 = S_{22} \quad (2.33)$$

$$S_{21} = \frac{2V_2}{Z_0 I} = S_{12} \quad (2.34)$$

$$\frac{\partial S_{11}}{\partial p} = \frac{\partial(\frac{2V_1}{Z_0 I} - 1)}{\partial p} = \frac{2\partial V_1}{Z_0 I \partial p} \quad (2.35)$$

$$\frac{\partial S_{21}}{\partial p} = \frac{2\partial V_2}{Z_0 I \partial p} \quad (2.36)$$

where p is the circuit parameter, V_l is the port voltage and Z_0 is the terminal reference impedance. Let us set $I = 1$ and $Z_0 = 50$. At each frequency point, we solve the system once to get all the S-parameters as well as their derivatives to circuit elements.

For the nonlinear optimization, gradient-based optimization algorithm, which is the same as neural network training algorithm, has been used to solve the nonlinear objective function. For a parameter to be constrained as $p_i \in (a_i, b_i)$, we choose

$$p_i^{new} = \begin{cases} a_i + e^{-|d_i|} (p_i^{old} - a_i) \\ b_i + e^{-\frac{p_i^{old} - a_i}{b_i - p_i^{old}} |d_i|} (p_i^{old} - b_i) \end{cases} \quad (2.37)$$

which maintains: $p_i^{new}|_{d_i=-\infty} = a_i$, $p_i^{new}|_{d_i=\infty} = b_i$, $p_i^{new}|_{d_i=0} = p_i^{old}$ and $p_i(d_i)$ is differentiable at $d_i = 0$. The benefits of the gradient-based optimization method are exponential learning step mechanism and circuit parameter range constraints.

Another optimization method, Genetic algorithm (GA), is used to extract the equivalent circuit parameters in [53]. Conventional optimization techniques such as the Levenberg-Marquardt (LM) method, which is used by SPICE for parameter extraction, are often subject to becoming trapped in local minimum. GA represents an effective method for determining the global minimum and less dependent upon the initial starting point.

An automated computer-aided generation of lumped element equivalent circuits for linear components such as resistors, capacitors, and inductors is introduced [54]. The method is based on a field theoretical analysis of distributed multiport circuit by the time-domain transmission-line matrix (TLM) method. The models provide an accurate

description of the distributed circuit within a specified frequency range of validity and exhibit a considerably reduced complexity.

2.4.3 Physical Based Equivalent Circuit Generation

In [48], an efficient method is introduced to extract equivalent circuit from full 3-D embedded girded structure. The procedure includes determining a set of fundamental building blocks for an embedded passive. These building blocks are used to construct a large area embedded passive that is geometrically comprised of many of the unit building blocks. Equivalent circuits of each of the building blocks are then extracted using a hierarchical extraction procedure. The objective passive's electrical behavior is based upon the behavior of each individual building block. The actual size of each block is chosen such that the physical dimension of the material being modeled is less than $1/10$ the wavelength of the maximum frequency of interest. The fundamental block equivalent circuit is achieved in an optimization procedure by measured or simulated S-parameters [55].

In [56], a general process for developing embedded capacitor model library is introduced. The library is a comprehensive library that combines EM simulation and test vehicles. The test vehicles are designed and tested to verify EM simulation accuracy. Ideally, EM simulation and test measurement results will agree with each other. However, factors such as material properties or physical dimensions that between EM simulation drawings and manufactured products can cause differences. Therefore, the combination of EM simulation and measurement working together provides a final library that is better than either EM simulation or measurement data alone.

Chapter 3

Proposed Neural Network Based Modeling

Techniques for Embedded Passives

3.1 Introduction

The wireless industry's emphases on time-to-market and low manufacturing cost, resulting from first-pass-design, are enhancing the demands on CAD tools for RF/microwave circuit design. ANN based models with physical/geometrical information and EM effects lead to an accurate and efficient CAD. Furthermore, the need for statistical analysis and yield optimization taking into account process variations and manufacturing tolerances in the components makes it extremely important that the component models are accurate and fast so that the design solutions can be achieved feasibly and reliably.

In this chapter, we first review pure neural network modeling approach for embedded passives. Then, we present a novel approach to model high-frequency effects of embedded passives in multilayer printed circuits based on combined equivalent circuit or state space theory together with neural networks [2]. Our combined model is a

hierarchical structure with two levels, motivated by Shirakawa, K., and et al [57]. In the lower level, a neural network maps the geometrical/physical parameters of the passive component into coefficient matrices of state equations or lumped component values of a user-defined equivalent circuit. In the higher level, we export the coefficient matrix into the state space equation or export the lumped component values into the equivalent circuit to compute the EM response in either frequency or time domain circuit design. The accurate and fast ANN based embedded passive models are trained from full wave EM data. The combined ANN based models can be directly used in time and frequency domain circuit simulators.

Our method combines existing modeling techniques and recent neural network approaches to efficiently perform simulation and optimization. Based on neural network techniques, geometry/physical parameters become design variables to improve circuit performance and reduce design and manufacturing cost.

3.2 Pure Neural Network Modeling Approach

Let $\mathbf{x}$ represent a N_x -vector containing parameters of a microwave device/component, e.g., length and width of an embedded resistor, or thickness and dielectric constant of an embedded capacitor. Let $\hat{\mathbf{y}}$ represent a N_y -vector containing the responses of the component under consideration, e.g., Y- or S-parameters. The EM/physics relationship between $\hat{\mathbf{y}}$ and $\mathbf{x}$ can be highly nonlinear and multi-dimensional. The theoretical model for this relationship may not be available, or theory may be too complicated to implement, or the theoretical model may be computationally too intensive for online microwave design and repetitive optimization (e.g., 3D full-wave EM analysis inside a Monte-Carlo statistical

design loop). We aim to develop a fast and accurate model by teaching/training a neural network to learn the embedded passive problem. Let the neural network model be defined as

$$\hat{y} = f_{ANN}(x, w) \quad (3.1)$$

where w represents the parameters inside the neural network also called as the weight vector. The most widely used neural network structure is the feedforward multilayer perceptrons (MLP) [4] [58] [59] where neurons are grouped into layers, and each neuron in a layer acts as a smooth switch that produces a response between low and high state according to the weighted responses of all neurons from the preceding layer. The neural network structure allows the ability to represent multidimensional nonlinear input/output mappings accurately, and to evaluate $\hat{y}$ from x quickly. To enable a neural network to represent a specific microwave x - $\hat{y}$ relationship, we first train the neural network to learn the microwave data pairs (x_i, d_i) where x_i is a sample of x , d_i is a vector representing the $\hat{y}$ data generated from microwave simulation or measurement under given sample x_i , and i is the sample index. For training purpose, we define an error function $E(w)$ as

$$E(w) = \frac{1}{2} \sum_{i \in Tr} \sum_{k=1}^{N_y} (f_{ANN_k}(x_i, w) - d_{ki})^2 \quad (3.2)$$

where d_{ki} is the k^{th} element of d_i , $f_{ANN_k}(x_i, w)$ is the k^{th} output of the neural network for input sample x_i , and Tr is an index set of all training samples. The objective of neural network training is to adjust neural network connection weights w such that $E(w)$ is minimized. For such a neural network, $\hat{y}$ is computed starting from the input layer $z_i^0 = x_i$, and then proceeding through the layers,

$$z_i^l = \sigma \left(\sum_{j=1}^{N_{l-1}} w_{ij}^l z_j^{l-1} + w_{i0}^l \right), \quad i = 1, \dots, N_l, l = 1, \dots, L \quad (3.3)$$

and finally $\hat{y}_k = z_k^L$. Here, x_i is the i^{th} input to the neural network, N_l is the number of neurons in layer l , z_i^l is the output of i^{th} neuron of l^{th} layer, w_{ij}^l represents weight of the link between j^{th} neuron of $l-1^{th}$ layer and i^{th} neuron of l^{th} layer, w_{i0}^l is the bias parameter of i^{th} neuron of l^{th} layer, and L is the total number of layers. The input and output layers are denoted as layer 0 and layer L . In Equation (3.3), $\sigma(\cdot)$ is the neuron activation function, which is usually a sigmoid function in the hidden layers and a linear function in the output layer.

A trained neural model can then be used online during microwave design stage replacing original slow model from EM simulators to provide fast model evaluation. The benefit of the neural model is especially significant when the model is repetitively used in design process such as optimization, Monte-Carlo analysis, and yield optimization [1] [60]. However, MLP models, trained to learn S-parameters data, cannot be used directly into time-domain circuit simulation and optimization. We aim to develop a fast and accurate combined model, which uses equivalent circuit and neural network, through EM data to learn the embedded passive problem.

3.3 Combined Equivalent Circuit and Neural Network (EC-NN) Modeling Approach

Embedded passives represent an emerging technology area that has the potential for increasing reliability, improving electrical performance, shrinking size, and reducing cost [61]. The conventional approach for circuit and system design requires equivalent circuits to capture the response of embedded passives. A number of fast equivalent circuit models of embedded passive components are available. In [48] [52], two methods are presented for developing equivalent circuit using optimization methods. Synthesis lumped element equivalent circuit from rational function of embedded passives is presented in [51]. Although we can get equivalent circuit in many ways from measured or simulated EM data, the lumped component values of the equivalent circuit are not directly related to geometrical structure of the embedded passive. If the embedded passive's geometrical parameters need to be changed, we have to re-generate a new equivalent circuit to match it.

3.3.1 Structure of the Combined EC-NN Model

Lets define $\mathbf{g}_p = \{R, L, C\}$, a N_p -vector containing the values of lumped or conventional components of a given equivalent passive circuit topology T_p . We use a neural network to represent $\mathbf{g}_p$ as

$$\mathbf{g}_p = f_{ANN}(\mathbf{x}, \mathbf{w}) \quad (3.4)$$

and then the combined model can be defined as

$$\tilde{\mathbf{y}}(\omega) = f_f(T_p(f_{ANN}(\mathbf{x}, \mathbf{w})), \omega) \quad (3.5)$$

$$y(t)=f_t(T_p(f_{ANN}(x,w)),t) \quad (3.6)$$

where ω is the angular frequency, $\tilde{y}(\omega)$ and $y(t)$ are the combined model response in frequency and time domain respectively, e.g., $\tilde{y}(\omega)$ can be S- or Y-parameters and $y(t)$ can be the currents $i(t)$ and voltages $v(t)$ of a two port embedded passive component. Therefore, a combined model realizes the $x - \tilde{y}/y$ relationship through a MLP and then the given equivalent circuit.

Our combined EC-NN model is a hierarchical structure with two levels. At the lower level, a neural network maps the geometrical/physical parameters into g_p vector. At the higher level, we insert the lumped component values into a given equivalent circuit to compute the EM response in frequency or time domain simulation. Fig. 3-1 shows the structure of the combined model for EC-NN.

For circuit CAD tools in time domain, we export our overall EC-NN model into SPICE sub-circuit format. The sub-circuit includes two parts, one is a set of functions to represent the internal structure of the low level MLP, which calculates the lumped component values based on different geometrical/physical parameters and insert them into another part, which is the topology of the given equivalent circuit.

Fig. 3-1: Structure of the combined equivalent circuit and neural network model illustrating the model development process and the testing phase. From top to bottom is the combined model training process, which includes parameter extraction and neural network training, and from bottom to top is the combined model feedforward calculation.

3.3.2 Combined EC-NN Model Development

We utilize an existing equivalent circuit and combine it with a MLP together to make the model automatically functioned in terms of geometrical/physical parameters. The EM data of embedded passives, which consists of geometrical/physical parameters (e.g., length and width of an embedded resistor) as inputs and real/imaginary parts of S-parameters as outputs, are generated by EM simulation or measurement.

To create data for MLP training, we extract the lumped component values based on the existing equivalent circuit through a set of measured/simulated sample pairs of EM data. Considering some noise in the EM data collected from measurement, the parameter extraction criteria for each set of input geometry is defined as an optimization objective function as

$$\underset{\mathbf{g}_p}{Min} \sum_{i \in T} \sum_{k=1}^{N_g} \|f_f(T_p(\mathbf{g}_p), \omega) - d_{ki}\|. \quad (3.7)$$

This objective function shows that the lumped component values $\mathbf{g}_p$ can be adjusted to make the S-parameters of high-frequency response of the equivalent circuit best match the EM data in the interested frequency bandwidth. Due to the complexity of the error function, iterative algorithms are used to explore the lumped component values. The optimization algorithms we used are gradient and quasi-Newton methods in Agilent-ADS [62]. To enforce the combined equivalent circuit model to be passive, we implement a constraint in the optimization process, such that all the lumped component should always be positive, $g_{pi} \geq 0$. We collect the lumped component values versus geometrical/physical parameters as neural network training data $(\mathbf{g}_{pi}, x_i)$. We teach/train a MLP to learn the relationship between equivalent circuit component values and geometrical parameters.

Let g_{pi} be a vector representing g_p data under given sample x_i . The error function is defined as

$$E(w) = \sum_{i \in Tr} \sum_{k=1}^{N_p} \|f_{ANNk}(x_i, w) - g_{pki}\| \quad (3.8)$$

where g_{pki} is the k^{th} element of g_{pi} . After training, the MLP can accurately calculate the component values varying with continuous geometry for the given equivalent circuit. The last step is to export the EC-NN model into a user defined simulator format, e.g., SPICE sub-circuit netlist format. The EC-NN model includes two sections. The first section is the trained neural network that calculates the lumped component values based on different geometry/physical inputs. The second section is the updated equivalent circuit, which receives the element values from MLP outputs. In a circuit simulator, the EC-NN model will be fed by geometrical/physical parameters as inputs. The MLP automatically calculates the element values in a user defined equivalent circuit and supply the values into the equivalent circuit to represent EM behavior in frequency and time domain.

3.4 Proposed Combined State Space Equation and Neural Network (SSE-NN) Modeling Approach

Topology of the equivalent circuit is a sensitive factor of the combined model accuracy. A given equivalent circuit's topology may not be suitable for different geometry and/or frequency bandwidth. However, the fixed equivalent circuit topology may not be accurate enough to reflect high frequency EM effects when the frequency range is extended or the geometry is changed abruptly. In order to develop an accurate

model, which can be represented more efficiently in both time and frequency domain simulation, we propose a combined SSE-NN modeling approach.

3.4.1 Formulation in Frequency Domain

EM data of an embedded passive can be collected based on different geometrical/physical parameters from full wave EM simulation/measurement. For a given frequency range, we can use transfer functions (polynomial rational functions) to represent the electrical behavior (e.g., admittance $\mathbf{Y}$ matrix) of the embedded passives. For any two-port embedded passive, the following three transfer functions are adequate to represent Y_{11} , Y_{21} , and Y_{22} , respectively.

$$H_1(s) = \frac{b_0 + b_1s + \dots + b_{n-1}s^{n-1} + b_ns^n}{a_0 + a_1s + \dots + a_{n-1}s^{n-1} + s^n} \quad (3.9.a)$$

$$H_2(s) = \frac{d_0 + d_1s + \dots + d_{n-1}s^{n-1} + d_ns^n}{a_0 + a_1s + \dots + a_{n-1}s^{n-1} + s^n} \quad (3.9.b)$$

$$H_3(s) = \frac{c_0 + c_1s + \dots + c_{n-1}s^{n-1} + c_ns^n}{a_0 + a_1s + \dots + a_{n-1}s^{n-1} + s^n} \quad (3.9.c)$$

where $s = j\omega$, and n is the number of effective order of the passive. Let us define a real coefficient vector $\mathbf{g}_v = \{a_0, a_1, \dots, a_{n-1}; b_0, b_1, \dots, b_n; c_0, c_1, \dots, c_n; d_0, d_1, \dots, d_n\}$. The relationship existing between the coefficients and geometrical/physical parameters would be highly nonlinear and very complicated. Therefore, we utilize neural network features to learn the highly nonlinear relationship between the coefficients of state space equation and geometrical/physical parameters.

In the coefficient parameter extraction procedure, we used gradient and quasi-

Newton optimization algorithms to enforce $H_1(s)$, $H_2(s)$, and $H_3(s)$ to best match EM data. The objective function is defined as

$$\text{Min}_{\mathbf{g}_v} \sum_{i \in Tr} \sum_{k=1}^3 \|H_k(\mathbf{g}_v, \omega) - d_{ki}\| \quad (3.10)$$

and we use a neural network to learn the relationship between coefficient vector $\mathbf{g}_v$ and EM input parameters $\mathbf{x}$,

$$\mathbf{g}_v = f_{ANN}(\mathbf{x}, \mathbf{w}). \quad (3.11)$$

We used the center point of input space as the initial point to optimize the coefficient vector values.

The frequency bandwidth of embedded passives, which will be used in time domain transient analysis, is related to the signal speed.

3.4.2 State Space Equation for Time Domain Simulation

Using coefficients $\mathbf{g}_v$, we can define

$$A = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 & \vdots & \vdots & \ddots & \vdots & \vdots \\ -a_0 & -a_1 & -a_2 & \cdots & -a_{n-1} & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & -a_0 & -a_1 & -a_2 & \cdots & -a_{n-1} \end{bmatrix}_{2n \times 2n}$$

$$B = \begin{bmatrix} 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 1 \end{bmatrix}_{2n \times 2}^T$$

$$C = \begin{bmatrix} b_0 - a_0 b_n & \cdots & b_{n-1} - a_{n-1} b_n & d_0 - a_0 d_n & \cdots & d_{n-1} - a_{n-1} d_n \\ d_0 - a_0 d_n & \cdots & d_{n-1} - a_{n-1} d_n & c_0 - a_0 c_n & \cdots & c_{n-1} - a_{n-1} c_n \end{bmatrix}_{2 \times 2n}$$

$$D = \begin{bmatrix} b_n & d_n \\ d_n & c_n \end{bmatrix}_{2 \times 2} \quad (3.12)$$

to form the state space equation,

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t) \end{cases} \quad (3.13)$$

where $\mathbf{x}(t)$ is a vector of internal states, $\mathbf{u}$ and $\mathbf{y}$ are vectors of the input and output signals, e.g., input voltages and output currents of the embedded passive respectively. Our combined models can be implemented into a time-domain circuit simulator using the state space equation (3.12) and (3.13) or into a frequency domain circuit simulator using (3.9).

Let us consider an example using a third order rational function to represent a 1-port embedded passive EM behaviour, as shown in (3.14).

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -a_0 & -a_1 & -a_2 \end{bmatrix}_{3 \times 3}$$

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}_{3 \times 1}^T$$

$$\mathbf{C} = \begin{bmatrix} b_0 & b_1 & b_2 \end{bmatrix}_{1 \times 3} \quad (3.14)$$

From the rational function for the one-port and third order embedded passive, the function can be expressed in state space equation format as,

$$i_2 = c_2 \cdot \dot{v}_2 = v_3$$

$$i_3 = c_3 \cdot \dot{v}_3 = v_4$$

$$i_4 = c_4 \cdot \dot{v}_4 = -a_0 v_2 - a_1 v_3 - a_2 v_4 + v_1$$

$$i_1 = b_0 v_2 + b_1 v_3 + b_2 v_4 \quad (3.15)$$

where v_1 is the port voltage, all of the capacitors (c_2, c_3, c_4) in equation (3.15) are equal to one unit, and v_2, v_3, v_4 are the effective internal states that are voltages of the unit capacitors, as shown in Fig. 3-2. The current value of voltage control current source (vccs) is $i_{c_k} = c_k \cdot \dot{v}_k$ ($c_k = 1, k = 2, 3, 4$), where $(k-1)$ is the total number of effective internal state for one-port model. The input voltage v_1 affects the node voltages of 2, 3, 4, and the output will be affected by all of the three internal nodal voltages.

Fig. 3-3 shows the general equivalent circuit of embedded passives based on 2 ports N^{th} order transfer function. The input voltages v_1 and v_2 affect the node voltages of 3, 4, ..., $N+2$ and $N+3, N+4, \dots, 2N+2$ respectively, and the outputs of the two ports are related to all of the $2N$ internal states.

Fig. 3-2: The general equivalent circuit of the one-port third-order embedded passive example.

$$i_1 = (b_0 - a_0 b_n) v_3 + \dots + (b_{n-1} - a_{n-1} b_n) v_{n+2} + (d_0 - a_0 d_n) v_{n+3} + \dots + (d_{n-1} - a_{n-1} d_n) v_{2n+2}$$

$$i_2 = (d_0 - a_0 d_n) v_3 + \dots + (d_{n-1} - a_{n-1} d_n) v_{n+2} + (c_0 - a_0 c_n) v_{n+3} + \dots + (c_{n-1} - a_{n-1} c_n) v_{2n+2}$$

Fig. 3-3: Demonstration of the general equivalent circuit structure, which is automatically generated in SSE-NN model, for a two-port n^{th} order ($n \geq 3$) embedded passives.

3.4.3 Stability and passivity

To assure that stability requirement in time domain simulation is satisfied, the poles of the combined SSE-NN model need to be on left half plane (LHP) of complex plane [63]. To enforce all the poles of the embedded passive transfer function, which are automatically generated from our combined SSE-NN model based on different geometrical/physical input parameters, to be in LHP, we add a set of constraints in the parameter extraction as,

$$P_{even-order} = \prod_{i=1}^T P_{2i} \quad (3.16)$$

where $P_{2i} = (s^2 + k_{2i}s + k_{3i})$ and $T = n/2$.

If $k_{2i} > 0$ and $k_{3i} > 0$, all of real and complex roots of P_{2i} are in LHP.

$$P_{odd-order} = P_1 \cdot \prod_{i=1}^T P_{2i} \quad (3.17)$$

where $P_1 = (s + k_1)$, $P_{2i} = (s^2 + k_{2i}s + k_{3i})$, and $T = (n-1)/2$.

If $k_1 > 0$, $k_{2i} > 0$ and $k_{3i} > 0$, all of real and complex roots of P_1 and P_{2i} are in LHP.

Let $k = \{k_1, k_{21}, k_{31}, \dots, k_{2T}, k_{3T}\}$ be a vector of components that leads to elements in the matrix A . For example, in a third order combined model, the denominator coefficients are defined as $a_0 = k_1 \cdot k_{31}$, $a_1 = k_1 \cdot k_{21} + k_{31}$, and $a_2 = k_1 + k_{21}$, respectively.

The criterion for passivity can be defined if the eigenvalues of $G = \text{Re}\{Y\}$ are positive. This condition can be assured if $y_{12}y_{21} \leq y_{11}y_{22}$, where the y_{jk} ($j, k = 1, 2$) are real parts of the Y matrix elements [63] [64]. It has been used as an optimization constraint in the g_v parameter extraction procedure for the combined SSE-NN model

development. This criterion is widely used in time domain model development, however, this is a necessary condition but not sufficient.

Another method, which uses convex programming approach to guarantee the combined models to be passive in time domain simulation, is presented in [65]. If $\mathbf{D} + \mathbf{D}^T > 0$, the positive definite condition is equal to requiring that there exists symmetric matrix $\mathbf{K} = \mathbf{K}^T \geq 0$ such that

$$\begin{bmatrix} -\mathbf{A}^T \mathbf{K} - \mathbf{K} \mathbf{A} & -\mathbf{K} \mathbf{B} + \mathbf{C}^T \\ -\mathbf{B}^T \mathbf{K} + \mathbf{C} & \mathbf{D} + \mathbf{D}^T \end{bmatrix} \geq 0 \quad (3.18)$$

is satisfied. This method can generate guaranteed passive time domain models, but it is very hard to implement into parameter extraction process.

The above criteria are added in the parameter extraction process to ensure that the rational functions not only accurately represent EM behavior but also enforce the time domain model to be stable and passive. We use ADS as the optimization tool to extract the coefficient values. The stable criterion and the criterion of $\text{Re}\{Y\}$ to be positive definite are conveniently implemented as constraints in the optimization procedure.

3.4.4 Structure of the Combined SSE-NN Model

Our combined SSE-NN model is a hierarchical structure with two levels. At the lower level, a neural network maps the geometrical/physical parameters into $\mathbf{g}_v$ vector. At the higher level, we insert the coefficient vector into the state equations to compute the EM response in frequency or time domain simulators. Fig. 3-4 shows the structure of the combined model SSE-NN. For circuit CAD tools in time domain, we export our SSE-NN into SPICE sub-circuit format. The lower neural network will be described by a set of

mathematical equations, which calculate the coefficient values based on different geometrical/physical parameters, and insert them into higher level. The equivalent circuit can be generated from (3.12) and (3.13).

3.4.5 Combined SSE-NN Model Development

EM data has component's geometrical/physical parameters and frequency as inputs and S-parameters as outputs. The next phase is parameter extraction, which is carried out for each geometry over the entire frequency range. The objective here is to determine the coefficient values that best fit the original EM data. Different geometrical parameter values and their corresponding coefficient values are then re-arranged into neural network training data. A 3-layer MLP neural network is trained using quasi-Newton algorithm in *NeuroModeler* [66]. For any given geometrical dimensions of the component within the range of the training data, the trained MLP can predict the elements of vector $\mathbf{g}_v$. We combine the state space equation with the neural model using our hierarchical setup to obtain the overall combined model. The inputs to the combined model are the geometrical dimensions of the embedded component. The intermediate outputs of the model are the corresponding coefficient vector values. The final outputs of the combined model are component's EM behavior, e.g., S-parameters. In the test phase, an independent set of test data containing S-parameters versus new geometrical parameter values (i.e., never seen during training) is generated using the EM simulator. This data is used to test the accuracy of the combined model. In the final phase, we formulate the combined model into a set of mathematical expressions to be directly used to carry out high-level circuit design in time-domain simulators.

Fig. 3-4: Structure of the combined state space equation and neural network model illustrating the model development process and the testing phase. Form top to bottom is the combined model training process, which includes parameter extraction and neural network training. From bottom to top is the combined model feedforward calculation.

3.5 Summary

Three neural network based modeling methods have been discussed in this chapter. The neural network based models are developed directly from EM data. In addition, they can speed up simulation and optimization in microwave circuit design. For different CAD environments and simulation/design purposes, Fig. 3-5 shows the three neural network based modeling methods for the embedded passives in time and frequency domain circuit design. Fig. 3-6 shows the flow chart of developing the neural based model for embedded passives in time and frequency domain circuit simulators.

Fig. 3-5: EM Based neural network modeling methods for embedded passives.

Fig. 3-6: Flow chart to demonstrate the three methods in modeling embedded passives for time and frequency domain circuit simulation.

Chapter 4

Application Examples of Neural Network based Models for Embedded Passives

In order to demonstrate the features of the three ANN based modeling methods, such as speed, accuracy, and efficiency, we use the pure neural network modeling approach, proposed combined EC-NN and SSE-NN modeling approaches to develop ANN based models for embedded resistors and capacitors. Since all of the three models are fast and accurate ANN based models, geometrical/physical parameters become design variables to improve circuit performance and reduce design/manufacture cost. We applied the pure neural network models to design a power amplifier circuit in a frequency domain circuit simulator. The SSE-NN models of embedded passives were used in signal integrity of multilayer circuit design in time domain circuit simulator. Optimization, Monte-Carlo analysis, and Yield-optimization were performed showing that the geometry inputs can be continuously adjustable by using our ANN based models and the models evaluation is much faster than computationally intensive physical/EM models of embedded passives in microwave circuit design.

4.1. Embedded Resistor

Accurate modeling of EM behaviors of embedded passives used in high-speed multilayer printed circuit boards is important for efficient high-speed circuit design. In this example, a model of a multilayer embedded resistor shown in Fig. 4-1 is developed. The embedded resistor is inserted between two perfect metal plane layers. The resistor is embedded in a FR4 substrate ($\epsilon_r = 4.1$) and the square resistivity (ρ) is $500\Omega/\text{mil}^2$ in the example. The EM data of the embedded resistor is automatically generated from planar EM simulation of Sonnet [67] by simulator driver in *NeuroModeler*. Simulator driver is a kind of interface program that can automatically call the EM simulator to solve the 3D microwave problem with user-desired geometrical/physical input parameters and automatically collect the S-parameters in the neural network training data format. Length (L), width (W) and frequency (f) are used as input parameters. The outputs are real and imaginary parts of S_{11} and S_{21} in EM data.

Fig. 4-1: 3-D physical structure of embedded resistor. Length and Width are adjustable variables in multilayer circuit design.

4.1.1 Embedded Resistor Example Using Pure Neural Network Model

The pure neural network model of embedded resistor is developed for using in an amplifier circuit example. The input parameters include length (L: 6 to 30 mils, step size 2 mils), width (W: 6 to 30 mils, step size 2 mils), and frequency (f: 0.1 to 1GHz, step size 0.1GHz; 1 to 20GHz, step size 1GHz). The total EM training data includes 5070 set of samples. We defined a MLP that includes three layers defined as input layer, hidden layer, and output layer, as shown in Fig. 4-2. The input layer has three neurons, the hidden layer includes eight neurons, and four neurons are in the output layer.

We trained the MLP in *NeuroModeler* using quasi-newton training algorithm. After 485 iterations, the training error achieved is 0.15% and testing error is 0.29%. The outputs are matched very well with testing data which was never used in training process. After training, the neural model can represent the S-parameters versus the variations of the three input variables.

The trained pure neural network model can be used directly in a frequency domain circuit simulator, e.g., *Agilent-ADS*, by using *NeuroADS*. *NeuroADS* is a kind of interface program that can plug our neural network model into a circuit schematic as a user-defined model in ADS. The model is a two-port black box and the outputs are S-parameters. Then, the EM effects will be calculated based on the neural network model's geometrical/physical input parameters. Therefore, the pure neural network models not only provide accurate and fast EM behavior in circuit design, but also have the geometrical parameters as design variables.

Fig. 4-2: The structure of 3-layers MLP Neural model for the embedded resistor. All of the neurons in the structure are fully connected.

In this example, a library of pure neural network models for embedded resistors is developed. The physical structure of the embedded resistors is the same as Fig. 4-1. However, we increased the dimensions of input parameters of embedded passives to five, called length (L), width (W), dielectric constant (ϵ_r), resistivity (ρ) and frequency (f), the model outputs are S-parameters [68]. The average number of training samples for each model in the library is 65,000 and the average training error is 0.5%.

4.1.2 Embedded Resistor Example Using the Proposed EC-NN Model

Fig. 4-3 shows the structure of the EC-NN model for the embedded resistor, which includes a user-defined equivalent circuit and a 3-layer MLP neural network. The embedded resistor model is going to be used in a signal integrity circuit example. Therefore, the embedded resistor input parameters are not exactly the same as the pervious resistor model. The EC-NN model's inputs are L (8 to 12 mils, step size 2 mils), W (35 to 55 mils, step size 2.5 mils), and f (0.1 to 1GHz, step size 0.1GHz; 1 to 16 GHz, step size 1GHz). First, we applied parameter extraction to collect the lumped components based on different geometry inputs over our interested frequency band in ADS. The passive condition ($g_{vi} \geq 0$) for each lumped component is easy to implement as the optimization constraints. Then, the MLP neural network is trained to learn the relationship between the geometrical inputs (L, W) and the four lumped component values (R1, R2, C1, C2) in *NeuroModeler*. The training data for the MLP neural network is shown in Table 4-I. After the MLP is well trained, it can accurately calculate the component values based on geometrical/physical input parameters for the given equivalent circuit even the input parameters have never been used in training.

Testing is performed by comparing the outputs of the overall EC-NN model and EM data, as shown in Fig. 4-4 and Fig. 4-5. Because the neural network can provide the accurate component values continuously varying with geometry for the equivalent circuit, the combined EC-NN model can be in place of the computationally intensive physical/EM model to efficiently provide EM effects in optimization and statistic design.

Fig. 4-3. The structure of the combined EC-NN model for embedded resistors. The equivalent circuit is a user defined circuit. The g_p is a vector that contains all of the lumped components in the user-defined equivalent circuit, as $g_p = \{R1, R2, C1, C2\}$.

The testing error of the combined EC-NN model is 5.8%. Further improvement of accuracy requires new topology of equivalent circuit. Instead of depending on understanding of physical structure, EM based experience, or human based trial and error process, we can use the proposed SSE-NN modeling method. As the equivalent circuit for the embedded resistor uses three capacitors, a 3rd order transfer function can express the behavior of the embedded resistor in the SSE-NN model.

Table 4-I. MLP training data, which is collected from parameter extraction, includes geometrical parameters and optimized lumped component values in the user defined equivalent circuit over the entire frequency range.

Width (mil)	Length (mil)	R2 (Ω)	R1 (Ω)	C1 (pF)	C2 (pF)
35.0	8	45.6683	8.6002	0.1275	0.0600
37.5	8	42.7001	7.9605	0.1302	0.0690
40.0	8	38.8894	6.8643	0.1406	0.0806
42.5	8	35.9236	6.7852	0.1479	0.0901
45.0	8	33.2270	6.7779	0.1559	0.0991
47.5	8	30.7799	6.8138	0.1643	0.1079
50.0	8	28.5589	6.8721	0.1729	0.1167
52.5	8	26.5397	6.9394	0.1821	0.1254
55.0	8	24.7006	7.0045	0.1924	0.1330
35.0	10	57.2060	9.1318	0.1513	0.0365
37.5	10	52.8035	8.5564	0.1591	0.0457
40.0	10	48.8439	8.1184	0.1671	0.0553
42.5	10	45.2734	7.7995	0.1754	0.0649
45.0	10	42.0413	7.5788	0.1841	0.0745
47.5	10	39.1032	7.4335	0.1933	0.0839
50.0	10	36.4203	7.3428	0.2021	0.0930
52.5	10	33.9601	7.2895	0.2127	0.1022
55.0	10	31.6958	7.2596	0.2229	0.1114
35.0	12	69.1841	10.9428	0.1682	0.0250
37.5	12	63.7455	10.2917	0.1785	0.0310
40.0	12	58.9087	9.7653	0.1889	0.0375
42.5	12	54.5952	9.3421	0.1994	0.0446
45.0	12	51.2501	9.0046	0.2102	0.0519
47.5	12	47.2370	8.7360	0.2210	0.0595
50.0	12	44.0631	8.5226	0.2325	0.0671
52.5	12	41.1570	8.3494	0.2440	0.0749
55.0	12	38.4795	8.2066	0.2557	0.0828

(a)

(b)

Fig. 4-4: Comparison of real and imaginary parts of S_{11} of embedded resistor EC-NN model outputs and independent EM data, which was never used in training. Curves A and B are generated based on $W = 53$ and $L = 11$ mils and $W = 39$ and $L = 9$ mils, respectively.

Fig. 4-5: Comparison of real and imaginary parts of S_{21} of embedded resistor EC-NN model outputs and independent EM data, which was never used in training. Curves A and B are generated based on $W = 53$ and $L = 11$ mils and $W = 39$ and $L = 9$ mils, respectively.

4.1.3 Embedded Resistor Example Using the Proposed SSE-NN Model

Table 4-II shows the model accuracy that we have achieved based on various orders of state equations in SSE-NN model development. Based on EM behaviors, to accurately represent embedded passives usually need no more than 4th order in rational function [50]. Therefore, we compared the accuracy for 2nd, 3rd, and 4th order rational functions in our combined model example. The table demonstrates that the optimal number of internal states is three. In the fourth order model, the additional internal state could not play an important role in the EM behavior representation. However, more coefficients are needed in transfer function and this leads to more freedom in parameter extraction and in neural network training. The training data for the MLP neural network, which is generated from parameter extraction based on 3rd order state space equation, is shown in Table 4-III.

The best results are obtained with the 3rd order SSE-NN model. The agreement between 3rd order SSE-NN model and EM data is achieved even though the independent testing data was never seen in training, as shown in Fig. 4-6 and Fig 4-7. To verify stability and passivity, the three LHP poles of the embedded resistor model at the geometry never used in training are shown in Table 4-IV, and the passivity condition is satisfied as shown in Fig. 4-8.

Table 4-II. Comparison of combined resistor model with different order formulation.

	Transfer Function	Test Error
2 nd Order	$H(s) = \frac{b_0 + b_1s + b_2s^2}{a_0 + a_1s + s^2}$	1.59%
3 rd Order	$H(s) = \frac{b_0 + b_1s + b_2s^2 + b_3s^3}{a_0 + a_1s + a_2s^2 + s^3}$	1.12%
4 th Order	$H(s) = \frac{b_0 + b_1s + b_2s^2 + b_3s^3 + b_4s^4}{a_0 + a_1s + a_2s^2 + a_3s^3 + s^4}$	2.38%

Table 4-III. The training data, which generated from parameter extraction based on 3rd order state space formulation, is used to train a MLP neural network in SSE-NN model for the embedded resistor. Each set of coefficient values are optimized based on each geometry of embedded resistor within the entire frequency range.

Width (mil)	Length (mil)	k1	k2	k3	b1	b2	b3	d1	d2	d3
35	8	4.43E-2	2.01E-1	9.18E-1	3.89E-4	6.92E-3	1.70E-2	-3.84E-4	-4.20E-4	9.69E-3
37.5	8	4.24E-2	2.05E-1	9.86E-1	4.00E-4	7.07E-3	1.89E-2	-3.94E-4	-4.46E-4	1.11E-2
40	8	3.80E-2	1.96E-1	9.90E-1	3.82E-4	6.72E-3	1.96E-2	-3.76E-4	-4.44E-4	1.19E-2
42.5	8	3.42E-2	1.86E-1	9.95E-1	3.66E-4	6.38E-3	2.02E-2	-3.59E-4	-4.18E-4	1.25E-2
45	8	3.09E-2	1.75E-1	1.00E+0	3.51E-4	6.03E-3	2.06E-2	-3.43E-4	-3.64E-4	1.30E-2
47.5	8	2.82E-2	1.64E-1	1.01E+0	3.39E-4	5.69E-3	2.09E-2	-3.30E-4	-2.82E-4	1.33E-2
50	8	2.55E-2	1.51E-1	1.00E+0	3.23E-4	5.28E-3	2.09E-2	-3.14E-4	-1.70E-4	1.32E-2
52.5	8	2.36E-2	1.40E-1	1.01E+0	3.16E-4	4.97E-3	2.12E-2	-3.05E-4	-2.80E-5	1.31E-2
55	8	2.16E-2	1.26E-1	1.00E+0	3.05E-4	4.55E-3	2.10E-2	-2.93E-4	1.48E-4	1.22E-2
35	10	4.28E-2	1.68E-1	8.92E-1	3.00E-4	6.77E-3	1.38E-2	-2.96E-4	-2.47E-5	6.76E-3
37.5	10	3.91E-2	1.68E-1	9.28E-1	2.94E-4	6.63E-3	1.56E-2	-2.89E-4	-7.87E-5	7.59E-3
40	10	3.33E-2	1.56E-1	8.88E-1	2.68E-4	6.04E-3	1.60E-2	-2.63E-4	-1.15E-4	7.98E-3
42.5	10	2.62E-2	1.35E-1	8.01E-1	2.24E-4	5.06E-3	1.56E-2	-2.19E-4	-1.61E-4	7.58E-3
45	10	2.12E-2	1.15E-1	7.22E-1	1.92E-4	4.30E-3	1.44E-2	-1.87E-4	-1.39E-4	7.19E-3
47.5	10	1.80E-2	1.03E-1	6.80E-1	1.73E-4	3.83E-3	1.39E-2	-1.67E-4	-1.26E-4	7.15E-3
50	10	1.49E-2	9.24E-2	6.19E-1	1.51E-4	3.36E-3	1.35E-2	-1.46E-4	-1.35E-4	7.18E-3
52.5	10	1.18E-2	7.75E-2	5.41E-1	1.27E-4	2.78E-3	1.23E-2	-1.21E-4	-1.13E-4	6.52E-3
55	10	9.22E-3	6.43E-2	4.65E-1	1.04E-4	2.26E-3	1.09E-2	-9.92E-5	-9.76E-5	5.77E-3
35	12	3.87E-2	1.49E-1	8.32E-1	2.27E-4	6.35E-3	1.22E-2	-2.23E-4	1.44E-4	4.35E-3
37.5	12	3.10E-2	1.40E-1	7.89E-1	1.95E-4	5.50E-3	1.40E-2	-1.91E-4	1.65E-5	4.57E-3
40	12	2.64E-2	1.25E-1	7.46E-1	1.78E-4	4.98E-3	1.35E-2	-1.74E-4	1.59E-5	4.85E-3
42.5	12	1.59E-2	8.79E-2	5.24E-1	1.14E-4	3.22E-3	1.08E-2	-1.11E-4	-6.05E-5	3.72E-3
45	12	1.15E-2	7.29E-2	4.25E-1	8.71E-5	2.51E-3	9.90E-3	-8.46E-5	-1.00E-4	3.54E-3
47.5	12	4.64E-3	4.54E-2	2.10E-1	3.70E-5	1.19E-3	6.92E-3	-3.59E-5	-1.64E-4	2.30E-3
50	12	4.21E-3	4.23E-2	2.08E-1	3.54E-5	1.13E-3	6.80E-3	-3.42E-5	-1.57E-4	2.44E-3
52.5	12	3.83E-3	3.95E-2	2.05E-1	3.40E-5	1.07E-3	6.71E-3	-3.26E-5	-1.49E-4	2.58E-3
55	12	2.58E-3	3.71E-2	2.04E-1	3.29E-5	1.02E-3	6.65E-3	-3.13E-5	-1.40E-4	2.71E-3

(a)

(b)

Fig. 4-6: Comparison of real and imaginary parts of S_{11} of embedded resistor SSE-NN model outputs and independent EM data, which was never used in training. Curves A and B are generated based on $W = 53$ and $L = 11$ mils and $W = 39$ and $L = 9$ mils, respectively.

(a)

(b)

Fig. 4-7: Comparison of real and imaginary parts of S_{21} of embedded resistor SSE-NN model outputs and independent EM data, which was never used in training. Curves A and B are generated based on $W = 53$ and $L = 11$ mils and $W = 39$ and $L = 9$ mils, respectively.

Table 4-IV. The three LHP poles of the 3rd order SSE-NN model.

EM Resistors	Root 1	Root 2	Root 3
$L(11) \times W(53)$	-6.1842e-2	-1.7428e-2+j0.16268	-1.7428e-2-j0.16268
$L(9) \times W(39)$	-1.0534e-1	-1.9736e-2+j0.21151	-1.9736e-2-j0.21151

Fig. 4-8: The 3rd order SSE-NN model in frequency-domain simulation. The W is swept from 35 to 55 mils. Let us define $D = y_{11}y_{22} - y_{12}y_{21}$, where y_{jk} ($j, k = 1, 2$) is real part of the Y matrix elements, the model is passive because D is always greater than zero.

4.2. Embedded Square Capacitor

The physical structure of an embedded square capacitor is shown in Fig. 4-9. We assume the material of the input/output connector is copper and capacitor is embedded in a FR4 substrate ($\epsilon_r = 4.1$) and the FR4 substrate is between two perfect metal layers. The thickness of capacitor is 0.3mil. The input parameters include length (L), capacitor dielectric constant (ϵ_{rcap}), and frequency (f). Real and imaginary parts of S-parameters are generated from 3D full wave EM simulator, Ansoft-HFSS [69]. For using in different examples, the geometrical and physical input parameters of the embedded capacitor models are in different ranges. Therefore, the pure neural network model and EC-NN/SSE-NN are trained from different EM data, which is generated from the same structure setup as shown in Fig. 4-9.

Fig. 4-9: 3-D physical structure of embedded capacitor. Length (L) and capacitor dielectric constant (ϵ_{rcap}) are adjustable variables in multilayer circuit design.

4.2.1 Embedded Capacitor Example Using Pure Neural Network Model

To develop a pure neural network model for the amplifier example, we assume the input space $\mathbf{x}$ includes length (L: 6 to 20 mils, step size 2 mils), capacitor dielectric constant (ϵ_{rcap} : 10 to 100, step size 10), and frequency (f: 0.1 to 1GHz, step size 0.1GHz; 1 to 20 GHz, step size 1GHz). The total training data includes 2400 set of EM samples. A 3-layer MLP is selected as shown in Fig. 4-10. We implemented ten neurons in the hidden layer of the MLP.

We train the MLP in *NeuroModeler* using quasi-newton training method. After 379 iterations, the training error achieved is 0.85% and the testing error reaches 1.04%. The outputs are matched very well with testing data which was never used in training process. After training, the neural model represents the S-parameters versus the variations of the three input variables.

We also developed a library of pure neural network models for embedded capacitors to cover a wide range of geometrical/physical input parameters. The models' input parameters are length (L), dielectric constant (ϵ_r), thickness (h), capacitor dielectric constant (ϵ_{rcap}) and frequency (f), and outputs are S-parameters [68]. The average of training samples for the pure neural network models in the library is 50,000 and the average training error is around 1%. Therefore, microwave circuit designers can get benefit from the model library to achieve fast and accurate EM behaviors for embedded passives in frequency domain circuit design loops.

Fig. 4-10: 3-layer MLP neural network model for the embedded capacitor.

4.2.2 Embedded Capacitor Example Using the Proposed EC-NN Model

The embedded capacitor input parameters are L (20 to 40 mils, step size 2 mils), ϵ_{rcap} (12.5 to 22.5, step size 5), and f (0.1 to 1GHz, step size 0.1GHz; 1 to 16 GHz, step size 1GHz). Fig. 4-11 shows the equivalent circuit used in our combined EC-NN model for the embedded capacitor. Parameter extraction is performed in ADS and passive condition is implemented by optimization constraints. After extracting the lumped component values versus different geometry inputs, the neural network is trained to learn the relationship between embedded capacitor's geometrical and physical inputs and lumped component values. For example, $L1 = 0.035155\text{nH}$, $C1 = 1.1354\text{pF}$, $C2 = 0.53687\text{pF}$ when the embedded capacitor's length = 26 mils and $\epsilon_{\text{rcap}} = 17.5$, and as capacitor's length = 30 mils and $\epsilon_{\text{rcap}} = 22.5$, $L1 = 0.037254\text{nH}$, $C1 = 1.4\text{pF}$, $C2 = 0.67104\text{pF}$.

The S-parameters comparison between the EC-NN model and original EM data is shown in Fig. 4-12 and Fig. 4-13.

Fig. 4-11: The combined EC-NN model structure for embedded capacitor. The equivalent circuit is user-defined. Let $g_p = \{C1, C2, L1\}$, a lumped component value vector is calculated by MLP in the combined EC-NN model.

(a)

(b)

Fig. 4-12: Comparison of S_{11} of embedded capacitor EC-NN model outputs and independent EM data, which was never used in training. Curves A and B are generated based on $L = 29$ mils, $\epsilon_{\text{rcap}} = 15$ and $L = 31$ mils, $\epsilon_{\text{rcap}} = 20$, respectively.

(a)

(b)

Fig. 4-13: Comparison of S₂₁ of embedded capacitor EC-NN model outputs and independent EM data, which was never used in training. Curves A and B are generated based on $L = 29$ mils, $\epsilon_{\text{rcap}} = 15$ and $L = 31$ mils, $\epsilon_{\text{rcap}} = 20$, respectively.

4.2.3 Embedded Capacitor Example Using the Proposed SSE-NN Model

Table 4-V illustrates the different accuracy that we have achieved based on various order formulas in SSE-NN model development. The optimal transfer function is 3rd order to represent the embedded capacitor. Testing is performed by comparing the outputs of combined SSE-NN models and EM data. The agreement between our 3rd order SSE-NN model and EM data is obtained even though the independent testing data was never seen in training, as shown in Fig. 4-14 and Fig. 4-15.

Table 4-V. Comparison of combined capacitor model with different order formulation.

	Transfer Function	Test Error
2 nd Order	$H(s) = \frac{b_0 + b_1s + b_2s^2}{a_0 + a_1s + s^2}$	2.20%
3 rd Order	$H(s) = \frac{b_0 + b_1s + b_2s^2 + b_3s^3}{a_0 + a_1s + a_2s^2 + s^3}$	1.67%
4 th Order	$H(s) = \frac{b_0 + b_1s + b_2s^2 + b_3s^3 + b_4s^4}{a_0 + a_1s + a_2s^2 + a_3s^3 + s^4}$	2.57%

(a)

(b)

Fig. 4-14: Comparison of S_{11} of embedded capacitor SSE-NN model outputs and independent EM data, which was never used in training. Curves A and B are generated based on $L = 29$ mils, $\epsilon_{\text{rcap}} = 15$ and $L = 31$ mils, $\epsilon_{\text{rcap}} = 20$, respectively.

(a)

(b)

Fig. 4-15: Comparison of S21 of embedded capacitor SSE-NN model outputs and independent EM data, which was never used in training. Curves A and B are generated based on $L = 29$ mils, $\epsilon_{\text{rcap}} = 15$ and $L = 31$ mils, $\epsilon_{\text{rcap}} = 20$, respectively.

4.3. Pure Neural Network Models of Embedded Passives in Frequency Domain Application

In this example, embedded passive components in an amplifier circuit are replaced by the pure neural network models trained using the above MLP models for embedded resistors and capacitors. The amplifier circuit, as shown in Fig. 4-16, is used for demonstration. Using neural networks, nonlinear relationships between device geometrical/physical parameters and corresponding electrical responses can be easily modeled. Monte-Carlo analysis and yield optimization of the circuit are performed by neural models trained from full wave EM data.

Fig. 4-16: The amplifier circuit, in which, all the passive components are represented by the pure neural network models. N_R and N_C denote neural models of embedded resistors and capacitors respectively.

N_{Cdc} and N_{Cd} are neural models of the decoupling capacitors of the amplifier. N_{Cc} 's are the neural models of coupling capacitors. N_{R1} , N_{R2} , N_{R3} , and N_{R4} represent neural models of the resistor bias networks, and N_{R5} is the neural model of the resistive load. A 3-D representation of the connections between different amplifier circuit components embedded in the substrate is shown in Fig. 4-17. As an example, the physical and/or geometrical parameters of two resistors and one capacitor are indicated in the figure. We constructed a schematic of the circuit in ADS. The responses of the circuit are obtained in the form of two-port S-parameters by performing small-signal AC analysis.

Fig. 4-17: 3-D illustration of physical connections between a few selected components in the amplifier circuit.

To optimize the gain of the amplifier circuit, a nominal optimization is first performed. The parameters considered in optimization are length and width of all the resistor MLP models. Table 4-VI shows the changing of geometry of embedded resistors after nominal optimization.

Table 4-VI. The geometrical inputs of embedded passives neural model before and after nominal optimization.

Design Variable	Before Optimization (mils)	After Optimization (mils)
L_1	9.0	9.6985
W_1	19.4	17.9367
L_2	14.0	14.7502
W_2	7.5	8.2589
L_3	25.0	27.8129
W_3	9.3	10.6673
L_4	16.5	19.2965
W_4	14.0	15.3901
L_5	30.0	28.2486
W_5	8.5	9.9595

After nominal optimization, let us consider the design specifications of the amplifier outputs are:

$$2.5 \text{ dB} \leq \text{Gain} \leq 4 \text{ dB}, \quad \text{for } f \leq 2 \text{ GHz}$$

$$-3 \text{ dB} \leq \text{Gain} \leq -1 \text{ dB}, \quad \text{for } 10 \text{ GHz} < f < 14 \text{ GHz}.$$

We assumed that L and W of embedded resistor have 10% deviation in manufacture process. Around the nominal optimization design centre, we performed yield analysis

for 200 random outcomes and the yield was 29%. In order to improve the yield, a yield optimization is carried out using the nominal design center as the starting point.

After yield optimization, the yield of the circuit is observed as 69%. The circuit responses after yield optimization are shown in Fig. 4-18. Table 4-VII shows that the geometrical parameters of resistors have been slightly moved around the design center values after yield optimization.

Table 4-VII. The comparison of geometrical inputs of embedded passives before and after yield optimization in the amplifier circuit.

Design Variable	Before Yield Optimization (mils)	After Yield Optimization (mils)
L_1	9.6985	9.8996
W_1	17.9367	16.9530
L_2	14.7502	15.2313
W_2	8.2589	8.6287
L_3	27.8129	27.0229
W_3	10.6673	10.2715
L_4	19.2965	19.7036
W_4	15.3901	15.5325
L_5	28.2486	27.8934
W_5	9.9595	10.0364

Fig. 4-18: Gain vs. frequency for *ADS* original circuit (—○—) and Monte-Carlo analysis using neural models (——).

4.4. The Proposed SSE-NN Models of Embedded Passives in Signal Integrity Circuit Example

To further confirm the validity of the proposed combined model in time-domain circuit design, we plugged the above embedded resistor and square capacitor 3rd SSE-NN models into a time-domain simulator, i.e., *Hspice* [70] to perform efficient circuit simulation and optimization including geometrical and physical parameters of the embedded passives. The combined models help to achieve a convenient link between EM behaviors and high-level circuit design, which improves design accuracy and efficiency. In this example, we used signal integrity of multilayer circuit as shown in Fig. 4-19, where the length (L) and width (W) of embedded resistor and length (L) and dielectric constant (ϵ_{rcap}) of embedded capacitor are adjustable. The input is a pulse signal, voltage equal to 5v and rise and full time are 0.1ns. In [71], the relationship between signal risetime and frequency bandwidth is presented as $f_{\text{BW}} = 0.35 / t_{\text{rise}}$. In our signal integrity example, the highest frequency component is 3.5GHz, our combined model covered 4 times f_{BW} . The transmission line model is defined in *Hspice* and more details about the circuit parameters can be found in Appendix A.

The optimization goal is to minimize the delay time and over shooting of the response signal. In optimization process, whenever optimization changes the geometry, the corresponding combined models are called with the new geometrical dimensions as inputs. From output comparison, as shown in Fig. 4-20, the output curves have been improved in terms of distortion and time delay.

Fig. 4-19: Three dimensional illustration of signal integrity of multilayer circuit with embedded resistor and capacitor.

Fig. 4-20: Comparison of signal from input buffer, and output signals before and after combined SSE-NN models optimization.

Table 4-VIII shows the models geometrical/physical values before and after optimization. The optimization used 136 iterations including repetitive evaluation of combined SSE-NN models to reach the criteria of the optimization goal. The total computation time based on our combined SSE-NN models was 3.75minutes. The result shows that the combined models provide possibility to efficiently adjust the geometry of embedded passives in high-frequency circuit design. Because we used neural models to learn the nonlinear relationship between geometry and coefficient vectors, the geometry becomes variable in circuit design.

Table 4-VIII. Comparison of the physical/geometrical parameters in embedded passives before optimization and after optimization.

		Before Optimization (Initial Value)	After Optimization (Design Centre)
Embedded Resistor	Length (mil)	11.50	8.50
	Width (mil)	45.02	52.50
Embedded Capacitor	Length (mil)	28.00	26.50
	ϵ_{rcap}	17.50	12.51

We also performed statistical analysis of the signal integrity example with our SSE-NN models in a three-coupled transmission line circuit as shown in Fig. 4-21. More details of the circuit information can be found in Appendix A. Monte-Carlo analysis of 500 signal integrity output curves with geometrical parameters as statistical design

variables is shown in Fig. 4-22. The total simulation time for 500 outcomes based on the geometry tolerance around the nominal design center is 8.24 minutes using proposed combined SSE-NN models by *Hspice*. However, the required time of Ansoft-HFSS for 500 different geometry is more than 8 hours. The proposed combined models retain the advantages of neural network model, such as speed and accuracy. They provide EM effects in high-level circuit design as well.

Fig. 4-21.: The three coupled transmission line circuit.

- : EM capacitor SSE-NN model;
- : EM resistor SSE-NN model

Fig. 4-22 : 500 output curves vs. time for *Hspice* Monte-Carlo analysis of the 3-coupled transmission line example using our combined SSE-NN models of embedded passives.

Chapter 5

Conclusions and Future Research

5.1 Conclusions

This thesis has addressed neural network based modeling methods for embedded passives in microwave/RF time and frequency domain circuit design. The pure neural network models, which can be trained efficiently with a good accuracy in *NeuroModeler*, have been used into frequency domain circuit simulator. For example, it can be easily plugged into *Agilent-ADS* using *NeuroADS*. However, since the pure neural network model is directly trained from S-parameters, it can not be directly used in time domain circuit simulators. The proposed combined modeling method makes it possible for the neural based model to be used in time domain circuit simulators.

The combined modeling method utilizes the existing modeling techniques, i.e., equivalent circuit or state space formulation, combined with neural network together to represent the EM effects in high frequency for embedded passive model development. The combined model is developed from S-parameters, which are collected from simulation or measurement, versus geometrical/physical parameters.

The accuracy of EC-NN model will depend on the equivalent circuit in the combined model for the entire frequency range. If an accurate and reliable equivalent circuit is

available, EC-NN will be generated efficiently, because the number of lumped elements in equivalent circuit is usually much less than the number of coefficient values in state space equations.

In SSE-NN model development, we do not need rely on understanding of physical structure, EM experience, and we can avoid human trial and error to find out a good equivalent circuit. An accurate general equivalent circuit is automatically generated for modeling embedded passives in SSE-NN method. Usually, the accuracy of empirical equivalent circuit model cannot be less than 5% because equivalent circuit is very sensitive to geometry variation and/or frequency bandwidth. SSE-NN modeling approach can achieve the average accuracy around 1.5% and fit for any geometry of embedded passives. The technique acts as a bridge to combine slow physical EM model and fast equivalent circuit model together. In high-speed digital circuit design, geometrical/physical parameters become design variables in circuit simulation and the combined model evaluates as accurate as EM model does and as fast as equivalent circuit model does. Therefore, manufacturing geometrical tolerance can be counted into circuit design.

5.2 Suggestions for Future Directions

Neural networks have been proved recently as a very powerful modeling technique for microwave devices and systems. From the research point of view, future works in complicated passive circuit modeling and automatic training algorithm will benefit the combined neural network model for development and application in all levels of microwave design such as modeling, simulation, optimization and statistical design.

An interesting topic following the idea in this thesis would apply the combined SSE-NN modeling approach to model complicated 3D passive circuit for microwave/RF circuit design.

One of the future works is to develop a library of combined neural network model for embedded passives in microwave circuit design. The library will provide efficient and accurate models to link commercial microwave simulator and EM simulators such that combined EM models can be used for circuit design and optimization.

Another interesting topic would be developing the combined model for embedded passives automatically. The algorithm should accurately predict how many orders are necessary and sufficient for the objective passive. The algorithm for automatic development of the combined model is to accurately predict the effective internal states of the embedded passives and the efficient number of hidden neurons in MLP that will help us to achieve the best model accuracy. After we achieved the user-desired accuracy, the automatic program can export the Spice format sub-circuit. Thus, we can use it directly in time domain circuit simulators.

Appendix A

Using Combined SSE-NN Models of Embedded Resistor and Capacitor for Optimization and Statistical Design in SPICE.

A.1 The Combined SSE-NN Model for Embedded Resistor

```
*****
** filename: Res3_new.inc
** Department of Electronics
** Carleton University
*****

***Resistor Transfer Function subnetlist***
.subckt Res3_new na nb Wid = 40 Len =10

*Neuromod input scaling
.param nmod_x1='-1.0+(2.0)*(Wid-(35.0))/((55.0) - (35.0))'
.param nmod_x2='-1.0+(2.0)*(Len-(8.0))/((12.0) - (8.0))'

*Neuromod calculating hidden neurons
.param nmod_z1 = '1.0 / ( 1.0 + exp(-1.0 *
(0.0624638+nmod_x1*(0.426305)+nmod_x2*(-0.366571))))'
.param nmod_z2 = '1.0 / ( 1.0 + exp(-1.0 * (-
3.28272+nmod_x1*(0.730402)+nmod_x2*(-1.64668))))'
.param nmod_z3 = '1.0 / ( 1.0 + exp(-1.0 *
(0.17974+nmod_x1*(0.286173)+nmod_x2*(1.40169))))'
.param nmod_z4 = '1.0 / ( 1.0 + exp(-1.0 * (-3.64692+nmod_x1*(-
0.850374)+nmod_x2*(1.26319))))'
.param nmod_z5 = '1.0 / ( 1.0 + exp(-1.0 * (1.40262+nmod_x1*(-
0.0586241)+nmod_x2*(1.14758))))'
.param nmod_z6 = '1.0 / ( 1.0 + exp(-1.0 *
(0.0797476+nmod_x1*(0.480045)+nmod_x2*(1.1474))))'
```

```

.param nmod_z7 = '1.0 / ( 1.0 + exp(-1.0 *
(1.93597+nmod_x1*(1.30044)+nmod_x2*(0.194225))))'
.param nmod_z8 = '1.0 / ( 1.0 + exp(-1.0 *
(0.240348+nmod_x1*(0.830603)+nmod_x2*(0.235417))))'

* Neuromod calculating output neurons
.param nmod_y1 = '1.65442+nmod_z1*(-
0.554404)+nmod_z2*(0.259146)+nmod_z3*(0.0521489)+nmod_z4*(3.25015)+n
mod_z5*(0.635561)+nmod_z6*(-1.37416)+nmod_z7*(-
1.56259)+nmod_z8*(0.707657)'
.param nmod_y2 = '-
1.20203+nmod_z1*(1.55599)+nmod_z2*(1.77441)+nmod_z3*(1.13855)+nmod_z
4*(4.66836)+nmod_z5*(1.78453)+nmod_z6*(-1.5311)+nmod_z7*(-
0.899452)+nmod_z8*(-0.418734)'
.param nmod_y3 =
'1.39358+nmod_z1*(1.0825)+nmod_z2*(1.36002)+nmod_z3*(1.14538)+nmod_z
4*(0.679454)+nmod_z5*(-1.3307)+nmod_z6*(-1.108042)+nmod_z7*(-
0.31076)+nmod_z8*(-0.663429)'
.param nmod_y4 = '-
0.407793+nmod_z1*(1.79694)+nmod_z2*(1.43898)+nmod_z3*(2.12983)+nmod_
z4*(2.10947)+nmod_z5*(0.810915)+nmod_z6*(-1.23471)+nmod_z7*(-
0.606733)+nmod_z8*(-1.94837)'
.param nmod_y5 = '-0.139262+nmod_z1*(-1.21543)+nmod_z2*(-
1.54717)+nmod_z3*(-0.612766)+nmod_z4*(-
0.657831)+nmod_z5*(0.988349)+nmod_z6*(0.533135)+nmod_z7*(0.325523)+n
mod_z8*(0.758474)'
.param nmod_y6 = '-
1.79565+nmod_z1*(2.97414)+nmod_z2*(2.98423)+nmod_z3*(0.37481)+nmod_z
4*(0.0148262)+nmod_z5*(0.927188)+nmod_z6*(0.422774)+nmod_z7*(-
0.31824)+nmod_z8*(-0.945675)'

*Neuromod output scaling
.param a0 = '0.0214341+(nmod_y1-(0.0))*((0.0421704) -
(0.0216898))/((1.0)-(0.0))'
.param a1 = '0.0295375+(nmod_y2-(0.0))*((0.061921) -
(0.0299442))/((1.0)-(0.0))'
.param b0 = '1.96982E-4+(nmod_y3-(0.0))*((3.63006E-4) - (2.01605E-
4))/((1.0)-(0.0))'
.param b1 = '0.00363469+(nmod_y4-(0.0))*((0.00599689) -
(0.00367728))/((1.0)-(0.0))'
.param d0 = '-3.44463E-4+(nmod_y5-(0.0))*((-2.03346E-4) - (-
3.72618E-4))/((1.0)-(0.0))'
.param d1 = '7.58975E-4+(nmod_y6-(0.0))*((0.00237567) - (7.58147E-
4))/((1.0)-(0.0))'

*Transfer Function Equation
e11 n11 0 vol = 'v(na)'
e21 n21 0 vol = 'v(nb)'
g81 0 n81 cur = '(-a0*v(n71)-a1*v(n81)+v(n21))'
c81 n81 0 lpf
r81 n81 0 1000000
g71 0 n71 cur = 'v(n81)'
c71 n71 0 lpf
r71 n71 0 1000000

```

```

g51 0 n51 cur = '(-a0*v(n41)-a1*v(n51)+v(n11))'
c51 n51 0 1pf
r51 n51 0 1000000
g41 0 n41 cur = 'v(n51)'
c41 n41 0 1pf
r41 n41 0 1000000

g21 nb 0 cur = 'd0*v(n41)+d1*v(n51)+b0*v(n71)+b1*v(n81)'
g11 na 0 cur = 'b0*v(n41)+b1*v(n51)+d0*v(n71)+d1*v(n81)'
.ENDS Res3_new

```

A.2 The Combined SSE-NN Model for Embedded Capacitor

```

*****
** filename: emc405.inc
** Department of Electronics
** Carleton University
*****
.subckt emc405 na nb Length =32 er=17.5

*Neuromod input scaling
.param nmod_x1='-1.0+(2.0)*(Length-(26.0))/((40.0) - (26.0))'
.param nmod_x2='-1.0+(2.0)*(er-(12.5))/((22.5) - (12.5))'

*Neuromod calculating hidden neurons
.param nmod_z1 = '1.0 / ( 1.0 + exp(-1.0 *
(5.16122+nmod_x1*(4.27541)+nmod_x2*(5.08521))))'
.param nmod_z2 = '1.0 / ( 1.0 + exp(-1.0 * (-
6.63584+nmod_x1*(2.8839)+nmod_x2*(-6.32316))))'
.param nmod_z3 = '1.0 / ( 1.0 + exp(-1.0 * (5.0922+nmod_x1*(-
1.43773)+nmod_x2*(3.6318))))'
.param nmod_z4 = '1.0 / ( 1.0 + exp(-1.0 * (-9.70603+nmod_x1*(-
2.18942)+nmod_x2*(-5.25676))))'
.param nmod_z5 = '1.0 / ( 1.0 + exp(-1.0 *
(6.13773+nmod_x1*(1.8852)+nmod_x2*(-2.97498))))'
.param nmod_z6 = '1.0 / ( 1.0 + exp(-1.0 *
(3.97025+nmod_x1*(3.52478)+nmod_x2*(4.24957))))'
.param nmod_z7 = '1.0 / ( 1.0 + exp(-1.0 *
(1.05621+nmod_x1*(0.153312)+nmod_x2*(0.108191))))'
.param nmod_z8 = '1.0 / ( 1.0 + exp(-1.0 * (25.0965+nmod_x1*(-
16.3275)+nmod_x2*(6.43237))))'

* Neuromod calculating output neurons
.param nmod_y1 = '9.13069+nmod_z1*(-
0.00418227)+nmod_z2*(1.42443)+nmod_z3*(3.86117)+nmod_z4*(1.14609)+nm
od_z5*(-0.239885)+nmod_z6*(-0.21753)+nmod_z7*(-8.26788)+nmod_z8*(-
6.01144)'
.param nmod_y2 = '10.6246+nmod_z1*(7.01201)+nmod_z2*(-
6.41685)+nmod_z3*(-13.6173)+nmod_z4*(9.66939)+nmod_z5*(-
2.35633)+nmod_z6*(-6.33211)+nmod_z7*(-3.68775)+nmod_z8*(7.54085)'

```



```

.param nmod_y3 =
'8.69456+nmod_z1*(0.233781)+nmod_z2*(1.74732)+nmod_z3*(4.51952)+nmod
_z4*(1.22997)+nmod_z5*(-0.108327)+nmod_z6*(-0.510698)+nmod_z7*(-
7.82511)+nmod_z8*(-6.65595)'
.param nmod_y4 = '10.5125+nmod_z1*(-
0.305085)+nmod_z2*(3.19532)+nmod_z3*(7.9875)+nmod_z4*(0.208544)+nmod
_z5*(-0.0123282)+nmod_z6*(0.00131921)+nmod_z7*(-9.56587)+nmod_z8*(-
10.6235)'
.param nmod_y5 = '3.74038+nmod_z1*(9.74575)+nmod_z2*(-
9.69266)+nmod_z3*(-21.0488)+nmod_z4*(10.9915)+nmod_z5*(-
3.31604)+nmod_z6*(-8.91162)+nmod_z7*(0.033856)+nmod_z8*(19.8931)'
.param nmod_y6 = '-11.1007+nmod_z1*(0.920695)+nmod_z2*(-
4.56163)+nmod_z3*(-11.2171)+nmod_z4*(0.813936)+nmod_z5*(-
0.0984896)+nmod_z6*(-0.594705)+nmod_z7*(11.284)+nmod_z8*(14.1626)'
.param nmod_y7 = '-0.0829401+nmod_z1*(-
9.22848)+nmod_z2*(9.41554)+nmod_z3*(20.5721)+nmod_z4*(-
10.1964)+nmod_z5*(3.18967)+nmod_z6*(8.43122)+nmod_z7*(-
1.06841)+nmod_z8*(-21.3238)'

*Neuromod output scaling
.param k1 = '2.06189+(nmod_y1-(0.0))*((4.79873) - (2.06189))/((1.0)-
(0.0))'
.param k2 = '6.61E-7+(nmod_y2-(0.0))*((0.0230037) - (6.61E-
7))/((1.0)-(0.0))'
.param k3 = '0.00306787+(nmod_y3-(0.0))*((0.0143045) -
(0.00306787))/((1.0)-(0.0))'
.param b1 = '0.0238229+(nmod_y4-(0.0))*((0.0729865) -
(0.0238229))/((1.0)-(0.0))'
.param b2 = '-0.0261649+(nmod_y5-(0.0))*((0.161306) - (-
0.0261649))/((1.0)-(0.0))'
.param d1 = '-0.0440381+(nmod_y6-(0.0))*((-0.0176996) - (-
0.0440381))/((1.0)-(0.0))'
.param d2 = '-0.112163+(nmod_y7-(0.0))*((0.044176) - (-
0.112163))/((1.0)-(0.0))'
.param b0 = 0
.param b3 = 1
.param d0 = 0
.param d3 = 1

.param a0 = '(k1*k3)'
.param a1 = '(k1*k2+k3)'
.param a2 = '(k1+k2)'

*Transfer Function Equation
e1 n1 0 vol = 'v(na)'
e2 n2 0 vol = 'v(nb)'
g8 0 n8 cur = '(-a0*v(n6)-a1*v(n7)-a2*v(n8)+v(n2))'
c8 n8 0 lpf
r8 n8 0 1000000
g7 0 n7 cur = 'v(n8)'
c7 n7 0 lpf
r7 n7 0 1000000
g6 0 n6 cur = 'v(n7)'
c6 n6 0 lpf

```

```

r6 n6 0 1000000
g5 0 n5 cur = '(-a0*v(n3)-a1*v(n4)-a2*v(n5)+v(n1))'
c5 n5 0 1pf
r5 n5 0 1000000
g4 0 n4 cur = 'v(n5)'
c4 n4 0 1pf
r4 n4 0 1000000
g3 0 n3 cur = 'v(n4)'
c3 n3 0 1pf
r3 n3 0 1000000
g2 nb 0 cur = '(d0-a0*d3)*v(n3)+(d1-a1*d3)*v(n4)+(d2-
a2*d3)*v(n5)+d3*v(n1)+(b0-a0*b3)*v(n6)+(b1-a1*b3)*v(n7)+(b2-
a2*b3)*v(n8)+b3*v(n2)'
g1 na 0 cur = '(b0-a0*b3)*v(n3)+(b1-a1*b3)*v(n4)+(b2-
a2*b3)*v(n5)+b3*v(n1)+(d0-a0*d3)*v(n6)+(d1-a1*d3)*v(n7)+(d2-
a2*d3)*v(n8)+d3*v(n2)'

.ENDS

```

A.3 The Optimization Netlist for Signal Integrity Example in SPICE

```

*****
* SPICE CIRCUIT FOR Optimization DEMO *
*****
. OPTION METHOD = GEAR
.options search = 'home/xding/spice/TF_model'
.INCLUDE res3_new.inc
.INCLUDE emc405.inc

.options list node post
VI1      In1 0      PWL      0.0ps 0V 1000ps 0.0v      1100ps      5V
          2900ps      5V 3000ps 0v      4000ps      0V

xcl out1 0 emc405
+Length=op(28, 26.5, 40)
+er=op(17.5, 12.5, 22.5)

xrl out1 0 res3_new
+Wid =op(45.02, 35, 52.5)
+Len = op(11.5, 8.5, 12)

ri3 In3 0 50
ro3 out3 0 50

ri2 In2 0 50

```

```

ro2 out2 0 50

m1 out out1 vdc vdc pch l=1u w=20u
m2 out out1 0 0 nch l=1u w=20u
vdc vdc 0 5v

.model pch pmos level=2
.model nch nmos level=2

cld out 0 1p

u1 In1 In2 In3 0 Out1 Out2 Out3 0 uel      L=0.027800

* Model for MICROSTRIP COUPLED LINES

.model uel U      LEVEL=3      plev=1      elev=1
      dlev=2
+ nl=3      ht=381u      wd=305u
+ th=25u      Sp=102u      ts=838u      KD=4.7

.tran 100ps 5000ps
.tran DATA=ibm OPTIMIZE=op RESULT=comp1 MODEL=optmod
.MODEL optmod OPT ITROPT=200 relin = 1e-5 relout=5e-6
.MEAS tran comp1 ERR1 par(r) v(out1) minval=1e-06 ignor=1e-07
.print V(out1)

.data ibm t r
1.00000n 0.
1.10000n 168.4m
1.20000n 1.1654
1.30000n 2.9287
1.40000n 4.5065
1.50000n 5.3695
1.60000n 5.7566
1.70000n 5.7785
1.80000n 5.6103
1.90000n 5.3731
2.00000n 5.1371
2.10000n 5.0024
2.20000n 4.8845
2.30000n 4.8580
2.40000n 4.8467

.tran 100ps 5000ps
.print tran V(In) V(out) V(out1)
.ENDdata

```

A.4 The Optimization Result from SPICE

optimization results

```

residual sum of squares      = 1.45976
norm of the gradient         = 113.001
marquardt scaling parameter = 339396.0
no. of function evaluations  = 191
no. of iterations            = 136

```

optimization completed

measured results < relout= 5.0000E-06 on last iterations

```

**** optimized parameters op
*
                                %norm-sen      %change

.param length      = 26.5000      $ 49.9619      2.8860n
.param er           = 12.5027      $ 49.9566      2.8950n
.param wid          = 52.5000      $ 38.4489m     20.7248u
.param len          = 8.5000       $ 43.0075m     -19.6275u
****

```

```

***** Star-HSPICE -- 2001.4   (20011215) 14:52:44 04/25/2002
solaris
*****
*****
***** operating point information      tnom = 25.000 temp=25.000
*****

```

A.5 The Netlist of Statistical Design for the 3-coupled

Transmission Line Example in SPICE

```

*****
• Spice Circuit for 3-Coupled transmission line      *
• in Monte-Carlo Analysis                          *
*****

.OPTION METHOD = GEAR
.options search = 'home/xding/spice/TF_model'
.INCLUDE res3_new.inc
.INCLUDE emc405.inc

.options list node post
VI1      In1 0      PWL      0.0ps 0V 1000ps 0.0v      1100ps      5V
          2900ps      5V 3000ps 0v      4000ps      0V

xc1 out1 0 emc405 Length=agauss(26.5, 1, 3) er=12.5
xr1 out1 0 res3_new Wid=agauss(52.5, 2.5, 3) len=agauss(8.5, 1, 3)

ri3 In3 0 50

```

```

ro3 out3 0 50

ri2 In2 0 50
ro2 out2 0 50

m1 out out1 vdc vdc pch l=1u w=20u
m2 out out1 0 0 nch l=1u w=20u
vdc vdc 0 5v

.model pch pmos level=2
.model nch nmos level=2

cld out 0 1p

u1 In1 In2 In3 0 Out1 Out2 Out3 0 ue1      L=0.027800

* Model for MICROSTRIP COUPLED LINES

.model ue1 U      LEVEL=3      plev=1      elev=1      dlev=2
+ nl=3          ht=381u      wd=305u
+ th=25u      Sp=102u      ts=838u      KD=4.7

.tran 100ps 5000ps Sweep monte=500
.print tran V(In) V(out) V(out1)
.End

```

Bibliography

- [1] X. Ding, B. Chattaraj, M.C.E. Yagoub, V.K. Devabhaktuni, Q.J. Zhang, "EM based statistical design of microwave circuits using neural models", *Int. Symp. on Microwave and Optical Technology (ISMOT 2001)*, Montreal, Canada, June, 2001, pp.421-426. (Invited Paper)
- [2] X. Ding, J.J. Xu, M.C.E. Yagoub, Q.J. Zhang, "A new modeling approach for embedded passives exploiting state space formulation", *European Microwave Conf. (EuMC 2002)*, Milan, Italy, Sept. 23-27, 2002. (Accepted)
- [3] Q.J. Zhang, M.C.E. Yagoub, X. Ding, D. Goulette, R. Sheffield, and H. Feyzbakhsh, "Fast and accurate modeling of embedded passives in multi-layer printed circuits using neural network approach", *Elect. Components & Tech. Conf.*, San Diego, CA, May 2002, pp. 700-703.
- [4] Q.J. Zhang, and K.C. Gupta, *Neural Networks for RF and Microwave Design*, Artech House (Norwood, MA, 2000).
- [5] A.H. Zaabab, Q.J. Zhang, and M.S. Nakhla, "A Neural network modeling approach to circuit optimization and statistical design", *IEEE Trans. Microwave Theory Tech.*, vol. 43, pp. 1349-1358, 1995.

- [6] P. Burrascano, S. Fiori, and M. Mongiardo, "A review of artificial neural networks applications in microwave computer-aided design", *Int. J. RF and Microwave CAE*, vol. 9, pp. 158-174, 1999.
- [7] F. Wang, and Q.J. Zhang, "Knowledge based neural models for microwave design," *IEEE Trans. Microwave Theory Tech.*, vol. 45, pp. 2333-2343, 1997.
- [8] R. Biernacki, J.W. Bandler, J. Song, and Q.J. Zhang, "Efficient quadratic approximation for statistical design", *IEEE Trans. Circuit Syst.*, vol. 36, pp. 1449-1454, 1989.
- [9] P. Meijer, "Fast and smooth highly nonlinear multidimensional table models for device modeling", *IEEE Trans. Circuit Syst.*, vol. 37, pp. 335-346, 1990.
- [10] Q.J. Zhang, F. Wang, and M.S. Nakhla, "Optimization of high-speed VLSI interconnects: A review", *Int. J. of Microwave and Millimeter-Wave CAD*, vol. 7, pp. 83-107, 1997.
- [11] G.L. Creech, B. Paul, C. Lesniak, and M. Calcaterra, "Artificial neural networks for accurate microwave CAD application", *IEEE Int. Microwave Symp.*, San Francisco, CA, 1996, pp. 733-736.
- [12] P.M. Waston and K.C. Gupta, "EM-ANN models for microstrip vias and interconnects in dataset circuits," *IEEE Trans. Microwave Theory Tech.*, vol. 44, pp. 2495-2503, 1996.

- [13] V.B. Litovski, J. Radjenovic, Z.M. Mrcarica and S.L. Milenkovic, "MOS transistor modeling using neural network," *Electronics lett.*, vol.28, pp.1766-1768, 1992.
- [14] V.K. Devabhaktuni, C. Xi and Q.J. Zhang, "A neural network approach to the modeling of heterjunction bipolar transistors from S-parameter data", *European Microwave Conf.*, Amsterdam, Netherlands, 1998, pp.306-311.
- [15] K. Shirakawa, M. Shimizu, N. Okubo and Y. Daido, "Structural determination of multilayered large signal neural network HEMT model", *IEEE Trans. Microwave Theory Tech.*, vol. 46, pp. 1367-1375, 1998
- [16] P. Burrascano, M. Mongiardo, C. Fancelli, and M. Mongiardo, "A neural network model for CAD and optimization of microwave filters", *IEEE Int. Microwave Symp.*, Baltimore, MD, 1998, pp. 13-16.
- [17] Y.H. Fang, M.C.E. Yagoub, F. Wang, and Q.J. Zhang, "A new macromodeling approach for nonlinear microwave circuits based on recurrent neural networks", *IEEE Trans. Microwave Theory Tech.*, vol.48, pp. 2335-2344, 2000.
- [18] P. M. Waston and K.C. Gupta, "Design and optimization of CPW circuits using EM-ANN models for CPW components", *IEEE Trans. Microwave Theory Tech.*, vol. 45, pp. 2515-2523, 1997.
- [19] C. Christodoulou, A.E. Zooghby, and M. Georgiopoulos, " Neural network processing for adaptive array antennas" *IEEE-APS Int. Symp.*, Orlando, FL, 1999, pp.2584-2587.

- [20] A. Veluswami, M.S. Nakhla, and Q.J. Zhang, "The application of neural network to EM-based simulation and optimization of interconnects in high speed VLSI circuits", *IEEE Trans. Microwave Theory Tech.*, vol. 45, pp. 712-723, 1997.
- [21] G. Kothapali, "Artificial neural networks as aids in circuit design", *Microelectronics J.*, vol.26, pp. 569-678, 1995.
- [22] Q.J. Zhang and M.S. Nakhla, "Signal integrity analysis and optimization of VLSI interconnects using neural network models", *IEEE Int. Circuits Syst. Symp.*, London, England, 1994, pp. 459-462.
- [23] T. Hong, C. Wang, and N.G. Alexopoulos, "Microstrip circuit design using neural networks", *IEEE Int. Microwave Symp.*, Atlanta, Georgia, 1993, pp. 413-416.
- [24] M. D. Baker, C.D. Himmel, and G.S. May, "In-situ prediction of reactive ion etch endpoint using neural networks" *IEEE Trans. Components, Packaging, and manufacturing Tech. Part A*, vol.18, pp. 478-483, 1995.
- [25] M. Vai, S. Wu, B. Li, and S. Prasad, "Reverse modeling of microwave circuits with bi-directional neural network models" *IEEE Trans. Microwave Theory Tech.*, vol.46, pp. 1492-1494, 1998.
- [26] M. Vai and S. Prasad, "Automatic impedance matching with a neural network", *IEEE Microwave Guided Wave Letter*, vol. 3, pp. 353-354, 1993.

- [27] F. Wang, V.K. Devabhaktuni, C.G. Xi, and Q.J. Zhang, "Neural network structures and training algorithms for RF and microwave application", *Int. J. RF and Microwave Computer-Aided Eng.*, vol. 9, pp. 528 – 556, 1999.
- [28] C. Cybenko, "Approximation by superposition of a sigmoid function", *Math. Control Signals Syst.*, vol. 2, pp. 303-314, 1989.
- [29] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximations", *IEEE Trans. Neural Networks*, vol. 2, pp. 359-366, 1989.
- [30] J. de Villiers and E. Barnard, "Backpropagation neural nets with one and two hidden layers", *IEEE Trans. Neural Networks*, vol. 4, pp. 136-141, 1992.
- [31] S. Tamura and M. Tateishi, "Capabilities of a four-layered feedforward neural network: Four layer versus three", *IEEE Trans. Neural Networks*. Vol. 8, pp.251-255, 1997.
- [32] J.J. Xu, M.C.E. Yagoub, and Q.J. Zhang, "Exact adjoint sensitivity for neural based microwave modeling and design", *IEEE int. Microwave Symp.*, 2001, pp. 1015-1018.
- [33] D.E. Rumelhart, G.E. Hinton, and R.J. Williams, *Learning internal representation by error propagation, Parallel Distributed Processing*, MIT Press, Cambridge, vol. I, pp.318-362, 1986.

- [34] S. Haykin, *Neural Networks: A comprehensive foundation*, Prentice Hall, Upper Saddle River, NJ, 1999.
- [35] K. Ochiai, N. Toda, and S. Usui, "Kick-out learning algorithm to reduce the oscillation of weights", *IEEE Trans. Neural Networks*, vol. 7, pp. 797-807, 1994.
- [36] S.J. Perantonis and D.A. Karras, "An efficient constrained learning algorithm with momentum acceleration", *IEEE Trans. Neural Networks*, vol. 8, pp. 237-249, 1995.
- [37] S. Haykin, *Neural Networks*, IEEE Press, New York, NY, 1994.
- [38] *Neural Network Toolbox: For use with Matlab*, The MathWorks Inc., Natick, Massachusetts, 1993.
- [39] R.A. Jacobs, "Increased rate of convergence through learning rate adaptation", *IEEE Trans. Neural Networks*, vol. 1, pp. 295-307, 1988.
- [40] R.L. Watrous, "Learning algorithms for connectionist networks: Applied gradient methods of nonlinear optimization", *Proc. IEEE Int. Conf. Neural Networks*, vol. II, San Diego, California, 1987, pp. 619-627.
- [41] D.B. Park, "Optimization algorithms for adaptive networks: Second order back propogation, second order direct propagation and second order hebbian learning" *Conf. Neural Networks*, vol. II, San Diego, California, 1987, pp. 593-600.

- [42] R. Battiti, "First and second-order methods for learning: Between steepest descent and newton's method", *Neural computation*, vol.4, pp.141-166, 1992.
- [43] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T.Vetterling, *Numerical Recipes: The Art of Scientific Computing*, Cambridge University Press, Cambridge, MA, 1986.
- [44] S. Haykin, *Neural Networks: A comprehensive foundation*, Prentice Hall, Upper Saddle River, HJ, 1999.
- [45] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi, "Optimization by simulated annealing", *Science*, vol. 220, pp. 671-680, 1983.
- [46] A.J.F. Rooij, L.C. Jain, R.P. Johnson, *Neural network Training Using Genetic Algorithms*, World Scientific, 1996.
- [47] N. Baba, Y. Mogami, M. Kohzaki, Y. Shiraishi, and Y. Yoshida, "A hybrid algorithm for finding the global minimum of error function of neural networks and its applications", *IEEE Trans. Neural Networks*, vol. 7, pp. 1253-1265, 1994.
- [48] R. Poddar and M.A. brooke, "Accurate high speed empirically based predictive modeling of deeply embedded gridded parallel plate capacitors fabricated in a multilayer LTCC process", *IEEE Trans. Advance Packaging*, vol. 22, pp. 26-31, 1999.

- [49] I. Timmins and K.L. Wu, "An efficient systematic approach to model extraction for passive microwave circuits", *IEEE Trans. Microwave Theory Tech.*, vol. 48, pp. 1565-1573, 2000.
- [50] K.L. Choi, N. Na and M. Swaminathan, "Characterization of embedded passives using macromodels in LTCC technology", *IEEE. Trans. Components, packaging, and Manufacturing Tech. Part B*, vol. 21, pp. 258-268, 1998.
- [51] K.L. Choi and M. Swaminathan, "Development of model libraries for embedded passives using network synthesis", *IEEE. Trans. Circuit and Syst.*, vol. 47, pp. 249-260, 2000.
- [52] J. Zhao, C. Frye, W.M. Dai, and L. Tai, "S parameter based experimental modeling of high Q MCM inductor with exponential gradient learning algorithm", *IEEE Trans. Components, packaging, and Manufacturing Tech. Part B*, vol. 20, pp. 202-209, 1997.
- [53] H. Yun and G.S. May, "Passive circuit model parameter extraction using genetic algorithms", *Elect. Components & Tech. Conf.*, San Diego, CA, Jun. 1999, pp. 1021-1024.
- [54] T. Mangold and P. Russer, "Full-wave modeling and automatic equivalent-circuit generation of millimeter wave planar and multilayer structure", *IEEE Trans. Microwave Theory Tech.*, vol. 47, pp. 851-858, 1999.

- [55] L.A. Carastro, R. Poddar, E.M. Martin, A. Brooke, and N.M. Jokerst, "Passive device modeling methodology using nonlinear optimization", *Int. Conf. Circuit and Syst.*, vol.6, Orlando, FL, 1999, pp. 53 - 56.
- [56] W. Blood, F. Ling, T. Myers, and M. Petras, "Library development process for embedded capacitors in LTCC", *Electrical Performance and Electronics Packaging Conf.*, Scottsdale, ZA, Oct. 2000, pp. 147-150.
- [57] K. Shirakawa, M. Shimizu, N. Okubo, and Y. Daido, "A large signal characterization of an HEMT using a multilayered neural network", *IEEE Trans. Microwave Theory Tech.*, vol. 45, pp. 1630-1633, 1997.
- [58] F. Wang, V.K. Devabhaktuni, and Q.J. Zhang, "A hierarchical neural network approach to the development of library of neural models for microwave design", *IEEE Trans. Microwave Theory Tech.*, vol. 46, pp. 2391-2403, 1998.
- [59] J. Bandler, M. Ismail, J. Rayas-Sanchez, and Q.J. Zhang, "New directions in model development for RF/microwave components utilizing artificial neural networks and space mapping," *IEEE APS Int. Symp.*, Orlando, FL, July 1999, pp. 2572-2575.
- [60] B. Chattaraj, M.C.E. Yagoub, X. Ding, Q.J. Zhang, "EM based optimization of the microwave circuits by neural models and their application to power distribution and decoupling optimization", *European Microwave Conf. (EuMC 2001)*, London, England, Sept., 2001, pp. 125-128.

- [61] K.L. Choi, N. Na, and M. Swaminathan, "Characterization of Embedded passives using macromodels in LTCC technology," *IEEE Trans. Component, packaging, and Manufacturing Tech.*, vol. 21, pp. 258-268, 1998.
- [62] *ADS v.1.5* Agilent Technologies, Palo Alto, CA, UAS, 2000.
- [63] S.H. Min and M. Swaminathan, "Efficient construction of two-port passive macromodels for resonant networks," *Proc. IEEE Int. Conf. Elec., Packaging*, Cambridge, MA. USA, 2001, pp. 229-232.
- [64] B. Gustavsen and A. Semlyen, "Enforcing passivity for admittance matrices approximated by rational functions," *IEEE Trans. Power Syst.*, vol. 16, pp. 97-104, 2001.
- [65] C.P. Coelho, J.R. Phillips, and L.M. Silveira, "A convex programming approach to positive real rational approximation", *Proc. IEEE Int. Computer Aided Design*, CA. USA, 2001, pp. 245-251.
- [66] *NeuroModeler v. 1.3*, Q.J. Zhang, Dept. of Electronics, Carleton University, Ottawa, Canada.
- [67] *Sonnet v. 7.0*, Sonnet Software, Inc. Liverpool, NY, USA.
- [68] X. Ding, M.C.E. Yagoub, and Q.J. Zhang, "Library development for embedded passives in multilayer PCB circuit", *presented in Nortel Networks for jointed research project meeting*, Ottawa, ON, Feb., 2001.
- [69] *Ansoft HFSS v.8.0*, Ansoft Corp., Pittsburg, PA, USA.

[70] *Hspice* v.2001.2, Avant! Corp., Fremont, CA, USA.

[71] *HSPICE User's Manual: Application and examples*, vol. III, Mate-Software Inc.,
Campbell, 1996.