INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

**Bell & Howell Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600**

UMI[®]

**Development of A Library of Neural Models for Microstrip Line
Components for High Frequency Circuit Design**

by

Homayun Feyzbakhsh, B.Sc.

**A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of
Master of Engineering**

Ottawa-Carleton Institute for Electrical Engineering

Department of Electronics

Carleton University

Ottawa, Ontario K1S 5B6

Canada

© Copyright August 1999, Homayun Feyzbakhsh

**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.

0-612-52389-6

Canada

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

**Development of A Library of Neural Models for Microstrip Line
Components for High Frequency Circuit Design**

submitted by
Homayun Feyzbakhsh, B.Sc.

in partial fulfillment of the requirements for the
degree of Master of Engineering

Q.J. Zhang, Thesis supervisor

J. Wight, Chair, Department of Electronics

Abstract

There is an increasing need for a fast method to solve problems in the microwave design area, and for a library of models which can characterize a large set of component behaviours (i.e., different outputs in terms of the inputs). Several approaches have been adopted to address the above mentioned problem, each of which has some disadvantages (i.e. fast methods lack the needed accuracy and accurate methods were not fast enough). In this work the neural network approach has been adopted. A library of neural models would provide a fast and accurate solution to the problem at hand and could be used at an on-line design stage to quickly solve a problem. The objective of this work is to demonstrate the capability and feasibility of large scale neural network and to apply the methodology for circuit level optimization.

Acknowledgements

The author gratefully acknowledges his indebtedness to Professor Q.J. Zhang for his guidance, support and patience throughout the course of this study at Carleton University.

The author also would like to especially thank Dr. Fang Wang for providing some of the technical guidance and software support for this work. My gratitude is also extended to all the staff and fellow graduate students in the Department of Electronics for their assistance and friendship. I am indebted to them for so much occupation of computing time. The author would also like to thank all graduate students in CAD group for their friendship.

Finally, I would like to express my gratitude and appreciation to my parents and sisters, whose years of support and encouragement gave me the opportunity to complete this work.

Contents

1	Introduction	1
1.1	Motivation of This Work	1
1.2	Objective of This Work	4
1.3	Thesis Structure	5
2	Literature Review	7
2.1	Recent Developments in Neural Network Applications to Microwave Circuit Analysis Problems	7
2.2	Neural Network Modeling: Problem Statement	10
2.3	Neural Network Structures	11
2.3.1	Standard Feedforward Neural Networks	12
2.3.2	Other Neural Network Structures	16
2.4	Neural Network Training	20
2.4.1	Objective	20
2.4.2	Different Types of Training Algorithms	21
3	Library of Neural Models for Microwave Components	31
3.1	Library Development: Problem Statement and Steps.	32
3.2	Microwave Component Neural Models.	35
3.2.1	Transmission Line Neural Models	35
3.2.2	Microstrip Bend Neural Models	51
3.2.3	Microstrip Symmetric T-Junction Neural Model.	64
3.2.4	Microstrip Open and Short Circuit Stub Neural Models.	72

3.2.5	Microstrip Radial and Butterfly Stub Neural Models.	82
3.2.6	Microstrip Width Taper Neural Models	93
4	Applications of the Library in Optimization and Yield	
	Analysis of Microwave Circuits	101
4.1	A Microstrip Filter	102
4.1.1	Simulation: Comparison with EM Simulations.	102
4.1.2	Optimization of Design	106
4.1.3	Yield Analysis.	108
4.2	A Microstrip Signal Splitter	111
4.2.1	Simulation: Comparison with EM Simulations.	113
4.2.2	Optimization of the Design	115
4.2.3	Yield Analysis.	117
4.3	Summary	120
5	Conclusions and Remarks	121

List of Figures

2.1	Structure of a neural network (Multilayer Perceptron)	13
3.1	Neural network structure for a single microstrip line.	38
3.2	A single microstrip line (a) 3 dimensional (b) 2 dimensional view of a single microstrip line along with its geometrical parameters.	39
3.3	Model accuracy comparison between different models.	41
3.4	Comparison between the simulator response and neural model response	42
3.5	The frequency response for single microstrip line with horizontal axis being the frequency and vertical axis being the S parameter of interest.	43
3.6	Neural network structure for a coupled microstrip line.	44
3.7	The geometric view of a coupled microstrip line (a) 3 dimensional (b) 2 dimensional view	46
3.8	Testing errors (vertical axis) for different samples (horizontal axis).. . . .	48
3.9	The neural model frequency response obtained for a coupled microstrip	

line with horizontal axis being the frequency and vertical axis being the relevant "S" parameter. The length is variable..	49
3.10 Geometrical view of a 90 degree bend (a) 2 dimensional (b) 3 dimensional..	52
3.11 Neural network structure for 90 degree bend..	53
3.12 The comparison between predicted and EM simulated results for a microstrip bend.	56
3.14 (a) 2 dimensional (b) 3 dimensional view of an arbitrary angle bend.	57
3.15 Neural network structure for arbitrary angle bend.	59
3.16 horizontal axis is the sample number and vertical axis is the relevant S parameter testing error.	61
3.17 Frequency response for different outputs of a variable bend junction with variable width with horizontal axis being the frequency and vertical axis is the relevant S parameter..	62
3.18 Average percentage testing error versus training samples for a) 90 degree bend and b) variable angle bend. The horizontal axis is the number of samples and the vertical axis is the testing error.	63
3.19 (a) two dimensional (b) three dimensional view of T-Junction.	66
3.20 Neural network structure for a symmetric T-Junction.	67
3.21 Neural predicted response (vertical axis) vs. EM simulated response.	69
3.22 The horizontal axis is the frequency and the vertical axis is the relevant S parameter. The output width is being varied.	70

3.23 Geometry of an open circuited stub.	74
3.24 Neural network structure for an open circuited stub.	75
3.25 Errors between neural prediction and actual testing data for an open circuit stub. The horizontal axis is the sample number and the vertical axis is the percentage testing error.	77
3.26 Frequency response obtained for output S parameters of an open circuit stub with different widths. The horizontal axis is the frequency and the vertical axis is the relevant S parameter.	77
3.27 (a) 2 dimensional (b) 3 dimensional view of short circuit stub.. . . .	78
3.28 Neural network structure for a short circuit stub.	79
3.29 Errors between neural prediction and actual testing data for a short circuited stub	81
3.30 Frequency response of a short circuited stub with horizontal axis being the frequency.	81
3.31 Radial stub (a) two dimensional b) three dimensional geometrical view. . .	84
3.32 Neural network structure for a radial stub.	85
3.33 Scattering for radial stub for (a) and (b) . The horizontal axis is the relevant simulated S parameter obtained from HP and the vertical axis is that obtained from neural prediction.	87
3.34 Frequency response of a radial stub with horizontal axis being the frequency and the vertical axis is the relevant S parameter. The length is being varied here.	87

3.35	A butterfly stub (a) 2 dimensional (b) 3 dimensional view	88
3.36	Neural network structure for a butterfly stub.	89
3.37	Testing errors of butterfly stub f. The horizontal axis is the sample number and the vertical axis is the percent testing error for the relevant S parameter.	91
3.38	Frequency response obtained for butterfly stub with the horizontal axis being the frequency and the vertical axis being the relevant S parameter.	92
3.39	Microstrip width taper Geometry (a) 2 dimensional (b) 3 dimensional view	94
3.40	Neural network structure for a width taper.	95
3.41	The horizontal axis is the sample number and the vertical axis is test error.	97
3.42	The width taper frequency response for the appropriate S parameters (vertical axis) obtained by neural model. W1 is being varied.	99
4.1	The schematic of a filter and a component neural model equivalent.. . . .	103
4.2	The comparison between neural network approach and EM approach where the triangles represent response from physical EM approach and the solid line represents the response from neural approach and the diamonds represent the equivalent model response.	105
4.3	The comparison between neural network approach and EM approach	

- where the triangles represent response from physical EM approach
and the solid line represents the response from neural approach
and the diamonds represent the equivalent model response.. 105
- 4.4 The optimized filter responses for (a) (dB) and b)(deg). The triangles
show the EM results whereas the solid lines show the neural network
approach and the diamonds represent the equivalent model response. The
vertical axis is the appropriate S parameter and the horizontal axis is
the frequency.. 107
- 4.5 The optimized filter responses . The triangles show the EM results
whereas the solid lines show the neural network approach and the
diamonds represent the equivalent model response. 107
- 4.6 The yield graphs for 50 iterations. 109
- 4.7 The yield graphs for (a)(dB) and in (b) (deg) for 50 iterations.
The vertical axis is the appropriate S parameter and the horizontal
axis is the frequency. 109
- 4.8 500 iterations of (a) (dB) and (b) (deg). The vertical axis is the
appropriate S parameter and the horizontal axis is the frequency. 110
- 4.9 500 iterations of (a) (dB) and (b) (deg). The vertical axis is the
appropriate S parameter and the horizontal axis is the frequency. 110
- 4.10 The schematic for a signal splitter and a component neural
model equivalent.. 112
- 4.11 Comparison between neural model approach versus EM approach.

- The triangles represent EM response and the solid line represents neural network curve and the diamonds represent the equivalent models response. The vertical axis is the appropriate S parameter and the horizontal axis is the frequency. 113
- 4.12 comparison between neural model approach versus EM approach.
- The triangles represent EM response and the solid line represents neural network curve and the diamonds represent the equivalent models response. The vertical axis is the appropriate S parameter and the horizontal axis is the frequency. 114
- 4.13. The optimized responses. The triangles represent the EM response and the solid line represent the neural network response and the diamonds represent the equivalent model response. The vertical axis is the appropriate S parameter and the horizontal axis is the frequency. 116
- 4.14 The optimized responses. The triangles represent the EM response and the solid line represent the neural network response and the diamonds represent the equivalent model response. The vertical axis is the appropriate S parameter and the horizontal axis is the frequency. . . . 116
- 4.15 50 trials shown yield analysis. The vertical axis is the appropriate S parameter and the horizontal axis is the frequency. 117
- 4.16 50 trials shown for (a) (dB) and (b) (deg). The vertical axis is the appropriate S parameter and the horizontal axis is the frequency. 118
- 4.17 500 iterations of (a) (dB) and (deg). The vertical axis is the appropriate

S parameter and the horizontal axis is the frequency..	119
4.18 500 iterations of (a) (dB) and (deg). The vertical axis is the appropriate	
S parameter and the horizontal axis is the frequency.	119

List of Tables

3.1	Parameter description for the entire library..	34
3.2	Inputs to single microstrip line neural model.	37
3.3	All trained neural models for single microstrip line are tested with one set of testing data.	40
3.4	Inputs to the coupled microstrip line model.	44
3.5	All trained neural models for coupled microstrip line are tested with one testing data.	47
3.6	Inputs to the neural 90 degree bend and their ranges.	54
3.7	All trained neural models for microstrip bend are tested with one testing data set.	54
3.8	Inputs to the arbitrary angle model	60
3.9	All trained neural models for arbitrary bend microstrip line are tested with one testing data.	60
3.10	Inputs to the T-junction neural model	67
3.11	All trained model for microstrip T-Junction (testing data set is almost of the same size as the training data set).	68
3.12	Inputs to the open circuit stub neural model.	75
3.13	All trained neural models for open circuited stub are tested with one testing data.	76
3.14	Inputs to short circuit stub neural model.	79

3.15 All trained neural models for short circuited stub are tested with one testing data set.	80
3.16 Inputs to radial stub neural model.. . . .	85
3.17 All trained neural models for radial stub are tested with one testing data set.	86
3.18 Inputs to butterfly stub neural model.	90
3.19 All trained neural models for butterfly stub are tested with one testing data set.	90
3.20 Inputs to width taper neural model.	96
3.21 All trained neural models for width taper are tested with one testing data.	96

Chapter 1

Introduction

In this chapter, a brief discussion of the problems with conventional CAD (Computer Aided Design) approaches and tools is provided. Then the neural approach, adopted in this work will be discussed and its advantages are compared with the conventional methods.

1.1 Motivation of This Work

The rapid development in electronics circuits and microwave engineering has resulted in large, complicated circuit design and analysis problems requiring intensively computational resources. The situation becomes even worse during the design cycle where the analysis should be repetitively performed. Optimization and yield analysis are good examples of such repetitive analysis. In yield analysis one or some of the input parameters are changed randomly within a certain range (to model the tolerance of the components parameters) to study the percentage of the units which are capable of meeting a certain objective. In optimization, repetitive analysis should be performed, changing one or some of the input parameters in each analysis to reach a certain goal. EM (Electromagnetic) simulations are computationally expensive especially when we are dealing with a VLSI chip with several interconnects. It can be impractical to solve the problem by typical EM

approach. The reason is that EM simulations need to solve partial differential equations, i.e., Maxwells' equations, which is a very slow process. Therefore performing the computations demands huge computational resources. Considering the fact that several parameters are changed during a design cycle, it is difficult to overcome the time-to-market challenge with a typical EM approach.

To overcome the time consuming nature of EM simulations and optimizations, many empirical models have been proposed for different microwave components. These models can be used for the component in use in the circuit and provide a faster response to the design problems. These models can be used where the accuracy is not a major concern, since these models only approximate the response within a certain range of the input space and the resulting responses are not as accurate as the EM simulations [1]. To carry out the simulation and optimization of the same component in another region, another approximate model should be made available and used. In other words, for each region, a separate model is necessary to cover the whole range of interest.

The method of looking up data in a table (i.e., the table look up method) can also be used to store the data to be used in the design. However, it requires a large storage memory especially if the input and output space are large since the storage increases exponentially with the dimensions of the input and output parameters of the component [2]. Besides, this method is unable to predict the output value for the input values between any two adjacent storage locations.

To address all the above mentioned problems, the neural network approach can be used to model the components [1][3]. This model is composed of multiple neurons. Each neuron takes the linear combination of all its inputs as the argument to a nonlinear func-

tion such as step function, sigmoid function, gaussian function,... and outputs the value of the function. Neural networks enjoy the advantage of learning from data and are capable of generalizing the data. This approach basically resembles human learning process. Neural networks have recently been used as a fast and flexible tool for modeling, simulation and optimization in microwave design[4][5][6]. Through a process called training, the neural network can learn and abstract from the microwave data. The trained neural models can be used during on-line design to provide instant solutions to the problem. These models can provide major increases in speed over the typical accurate EM analysis of microwave design especially when the problem is solved by wave equations. This fast response enables designer to change the parameters several times during a design cycle and see the result of interest instantly. As will be seen later, neural models are as accurate as EM models, so there is no trade off between speed and accuracy in the neural network approach.

Optimization is a very important step in microwave circuit design towards a first pass success[1]. Usually the output response of interest is not always what is required from the specification and some input parameters should be changed to reach the required goals. Every microwave design software provides optimization capability which depends heavily on the models provided for the components. For the conventional CAD approaches, the more accurate the model is, the longer it takes to provide the response. The problem is even worse with large scale engineering optimization problems. With the neural network approach, the optimization can be done quickly and as accurately as the original EM optimizations since the neural network is capable of generalization.

Yield analysis is another challenge in the microwave design area[3]. The objective of this kind of design is to study the effect of parameter tolerances on the outputs of inter-

est to see how suitable the design is for production and how many of the produced units violate the minimum specification. Several circuit analyses need to be carried out corresponding to different input parameters within their tolerances to imitate the production process. To investigate the use of neural models in yield analysis is another motivation for this work.

1.2 Objective of This Work

The first objective of this work is to develop a library of neural models for commonly used microwave components such as single and coupled transmission line, transmission line bend, T-Junction, open and short circuited stub, radial and butterfly stub and microstrip width taper. Neural model should be capable of providing the same behavior (i.e., S parameters) of components, as those obtained by EM simulation within an acceptable error range.

The training data set is obtained by HP Momentum analysis[7] where the moment method (which is the accurate analysis) is utilized to provide the required response through the EM simulation. This set is used to train the model by NeuroModeler [8]. Then the model obtained is tested with testing data obtained by momentum. To ensure that the neural model response was due to the learning, and not memorization of the data samples, the testing data set is independent from the training data set.

Another objective is to use the library in microwave circuit design to judge its accuracy and to investigate the further use of neural models in real-world applications such as yield analysis and optimization. The improvement offered by neural approach

becomes more evident with the examples given in Chapter 4.

1.3 Thesis Structure

This thesis is divided into five chapters as follows:

In Chapter 2, a literature review will be presented including the current status of neural networks, with emphasis on neural network structures that are suitable for microwave applications. A description of neural structures such as standard feed forward networks will be provided. A review of the training methods used in microwave applications such as backpropagation, second-order algorithms and decomposed optimization algorithms will be conducted. Other training methods such as global training algorithms will also be briefly described. The second-order training algorithms and back propagation techniques are widely used for training purposes. However, the problem with these techniques is that they can't escape from the local minima. Global training algorithms such as simulated annealing allows the optimization to move towards another local minimum.

In Chapter 3, a library of microwave components is first defined. Accurate and efficient electromagnetically trained artificial neural network models have been developed for this library and the results have been provided for the following microwave components: 90 degree bend, arbitrary angle bend, single and coupled transmission lines, open circuit stub, short circuit stub, microstrip radial stub, microstrip butterfly stub, symmetric T-Junction and microwave width taper. The training and testing results were presented for the developed models. The training was conducted with MLP structure and second order methods for different number of hidden neurons and different training sample sizes to find

the best model. All these results are shown in appropriate tables.

In Chapter 4, two circuit design examples are given to show the validity of the neural models developed in Chapter 3. In the first example, a microstrip filter is designed. The filter was simulated by using the neural models obtained in the previous chapter and the resulting responses were compared to those obtained from electromagnetic simulation. Then an optimization was performed to obtain the required frequency response. Yield analysis was performed afterwards, using neural models to investigate how many of the units can meet the requirements after the optimization. The second example is to design a signal splitter which divides the signal into two different paths. Again the response of the circuit using the neural model components was compared to that of the EM simulations. Optimization and yield analysis are performed afterwards.

Finally, in Chapter 5, conclusions and remarks are presented.

Chapter 2

Literature Review

2.1 Recent Developments in Neural Network Applications to Microwave Circuit Analysis Problems

As a result of rapid developments in electronics, electronic analysis and design have become a complicated, large scale problem with many interconnects and components. Solving these problems requires highly efficient computational systems. For instance, modeling, optimization and yield analysis are the most important stages in the circuit design process. However, the complex models such as EM models for microwave components, when used with highly repetitive analysis, are computationally demanding. For yield optimization, where an even larger number of circuit analysis is required, the computational complexity is much higher. Current design tools for microwave circuits suffer from a lack of accurate and efficient models for components. Accurate and efficient models are essential for accurate and instant solutions, therefore, fast and accurate design tools are required to address today's time-to-market challenge. Analysis and optimization tools with EM models of microwave components provide highly accurate solutions, how-

ever these models are highly computationally intensive, and consequently, very slow which makes them unsuitable choices for repetitive analysis used in interactive CAD. Equivalent circuit models are fast to provide a response, but they are based on some assumptions and therefore have some limitations on the range of use. In addition to all these restrictions, empirical models lack the required accuracy. The table-look-up method is generally valid only for the structures measured on the grids, so the building of the table is very time consuming and its size increases exponentially with the number of variables.

To address the above mentioned challenges, a new CAD approach based on a neural network approach has been introduced in recent years for microwave modeling [1][3][4][5][8]. These models are trained off-line and can be used on-line during the design cycle providing instant solutions to the original EM problem. Neural networks are used as trainable devices with the ability to generalize. A neural network model can be developed by learning and abstracting from a simulated EM problem through a process called training. When the training is over, the model can be used during microwave design to provide fast and accurate solution to the original problem.

Recently a number of microwave components have been accurately modeled by neural networks such as FET devices [1][4][9][10], spiral inductors [8][11], vias [12][13] and power transistors and amplifiers [14], coplanar waveguide (CPW) circuit components [15] and interconnects [16]. In these approaches, Multilayer Perceptrons (MLP) are usually used to represent the complicated input/output relationship in the problem under study. The same approach has been adopted for circuit simulation and optimization [12][17], signal integrity analysis and optimization of VLSI interconnects[16][19], filter design [20], IC modeling [9], process design [21], synthesis[5], Smith chart representation

[22] and microwave impedance matching [23]. These works have established the foundation of neural modeling techniques in microwave design.

Neural models are much faster than EM models [1] [3] while they are more accurate than empirical models [24] and can handle more dimensions than table-look-up models [2] and can be developed more easily once a new technology is introduced [18].

The accuracy of neural models depends on the training data presented to it. Training data should be obtained with sufficient accuracy and well distributed to obtain accurate models. Generating accurate training data can be costly depending on the accuracy required. So there is always a trade off between the required accuracy of the training data and the required time to produce it. Another major factor in the accuracy of the neural model is the choice of the neural structure since a problem and especially a microwave problem may be nonlinear, non smooth, or contains many variables. With the right choice of neural structure, a higher accuracy may be reached with fewer training data. For instance, to model a smooth, slowly varying nonlinear function, a feedforward neural network with smooth switching function is a smart and efficient choice. However, to model a highly nonlinear function a feedforward neural network with gaussian functions in the hidden layer is more efficient. An efficient model also should have the optimum number of hidden neurons since a network with too small a number of hidden neurons is unable to learn and represent the problem accurately. On the other hand, a neural network with too large a number of hidden neurons leads to overlearning.

Training is an important part of model development, since it is not possible to get an accurate model even with the right structure unless it is trained by the right training algorithm. A problem usually faced in training is the existence of many local minima. Glo-

bal optimization techniques can be used to overcome this problem. Once the neural model is obtained, it can be used in repetitive circuit analysis and optimization to replace the original and slow EM simulation. The problem statement of neural modeling will be described in the following section and some useful neural network structures will be reviewed, with emphasis on the MLP structure.

2.2 Neural Network Modeling: Problem Statement

Assume that $\mathbf{x}$ is a vector of parameters of a given component, such as the width or the length of a microstrip line and $\mathbf{y}$ is a vector of the required responses, such as the S parameters of the microstrip line. The relationship between $\mathbf{x}$ and $\mathbf{y}$ can be, and usually is nonlinear and multidimensional in an EM problem. This relationship can be represented by a neural network once trained with a set of $\mathbf{x}$ - $\mathbf{y}$ sample pairs. This set of sample pairs is called training data and is generated from the original EM simulations or measurements (in our case, ADS Momentum analysis was used to create training data to train the models for microwave components, e.g., single and coupled transmission lines, open and short-circuited stub, and so on). The neural network model for this relationship can be represented by the equation

$$\mathbf{y} = \mathbf{y}(\mathbf{x}, \mathbf{w}) \quad , \quad (2.1)$$

where $\mathbf{w}$ is the weight vector parameter of a neural network model parameter, $\mathbf{x}$ and $\mathbf{y}$ are the inputs and outputs of the neural model. The definition of $\mathbf{w}$ and the way $\mathbf{y}$ is defined using $\mathbf{x}$ and $\mathbf{w}$ defines the structure of the neural network. Training is the process during which $\mathbf{w}$ is determined such that the error between neural outputs and desired outputs

(provided by training data from EM simulation) is minimized. This error is defined as [47]:

$$E = \frac{1}{2} \sum_{p=1}^{N_p} \sum_{k=1}^{N_k} (y_{pk}(x_p, \mathbf{w}) - d_{pk})^2, \quad (2.2)$$

where d_{pk} is the k^{th} element of the desired output vector $\mathbf{d}_p$ and $y_{pk}(x_p, \mathbf{w})$ is the k^{th} output of neural network when the input x_p is provided to the model. N_p and N_k represent the number of outputs and the number of samples. Once trained, the neural model can be used for predicting the output values when the input vector is presented. The model is then tested with an independent set of input-output samples (which is called testing data) to test the accuracy of the model. After testing, the model can be used during the on-line design process to repetitively provide instant solutions when the input parameters change. The enhancement of neural models in microwave is especially noticeable when the model is repetitively used during a design process such as Monte Carlo analysis.

2.3 Neural Network Structures

In this section, different structures of neural networks will be discussed. Each structure realizes the function $y = y(x, \mathbf{w})$ in a different manner since the way $\mathbf{w}$ is defined and y is obtained from $\mathbf{x}$ and $\mathbf{w}$ is different for each structure. Emphasis is on standard feedforward neural networks, since these structures are more popular in microwave applications. A brief introduction about MLP and radial basis function (RBF) networks is first provided. Then some other neural structures such as Knowledge Based Neural Net-

works with its variations, wavelet networks and projection pursuit learning networks will be discussed.

2.3.1 Standard Feedforward Neural Networks

The standard feedforward neural network is basically capable of approximating generic functions, including continuous functions. Multilayer Perceptrons (MLP) are among the most important and widely used types of feedforward network architectures. An MLP neural network consists of an input layer, one or more hidden layer(s) and an output layer, as shown in Figure 2.1. This is an I+2 layer MLP.

The number of hidden layers is I. Let the number of neurons in hidden layer i be N_i , $i=1,2,\dots,I$, w_{kj}^i represent the weights of the link between k^{th} neuron of $i-1^{th}$ hidden layer and k^{th} neuron of i^{th} hidden layer, θ_k^i be the bias parameter of the k^{th} neuron of the i^{th} hidden layer and x_i be the i^{th} input. $\bar{y}_k^i$, the output of the k^{th} neuron of the i^{th} hidden layer can be computed according to standard MLP formulae as [47],

$$\bar{y}_k^i = \sigma\left(\sum_{j=1}^{N_{i-1}} w_{kj}^i \cdot \bar{y}_k^{i-1} + \theta_k^i\right), \quad (2.3)$$

where $k = 1, \dots, N_i$, $i = 1, \dots, I$, $y_k^1 = x_k$, $k = 1, \dots, N_x$, $N_x = N_0$, $\sigma(\cdot)$ is called activation function and is a sigmoid function given by

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

This is not the only activation function. Other candidates are the inverse tangent

Figure 2.1: Structure of a neural network (Multilayer Perceptron) N_x - N_z -...- N_I - N_y , where N_x is the number of input neurons, N_y is the output of output neurons and N_I to N_I are the number of hidden neurons.

function given by

$$\sigma(t) = \arctg(t) , \quad (2.5)$$

and the hyperbolic tangent function given by:

$$\sigma(t) = \frac{e^t - e^{-t}}{e^t + e^{-t}} . \quad (2.6)$$

All these functions are bounded, continuous and monotonic. If v_{ij} represents the weight of the link between i^{th} neuron of output layer and j^{th} neuron of I^{th} hidden layer and β_i is the bias parameter of the i^{th} output, the outputs of MLP can be expressed as:

$$y_i = \sum_{j=1}^{N_I} v_{ij} \bullet \bar{y}_j^I + \beta_i , \quad i = 1, \dots, N_y . \quad (2.7)$$

This is a linear expression for output neurons. Training parameters consist of $w_{kj}^i, j = 1, \dots, N_{i-1}$ and $\theta_k^i, (k = 1, \dots, N_i), i = 1, \dots, I$ and $v_{ki}^I, (i = 1, \dots, N_I),$ where $(k = 1, \dots, N_y), \beta_k, (k = 1, \dots, N_y)$. A two layered MLP with no hidden layer is not capable of approximating nonlinear continuous functions. According to the universal approximation theorem, a three layered MLP with one sigmoidal hidden layer can theoretically model any real function with any degree of complexity, if there are enough hidden neurons available [26]. However, in reality a neural network with up to 4 layers and a finite number of hidden neurons (less than 100) can be used to model the microwave devices and

circuits. It can be shown that a four layered MLP is preferred in the applications where a behavioral repetition exists in the output space [4]. for their generalization capability, since the four layered MLP's perform better in defining the decision boundaries in pattern classification tasks.

Feedforward neural networks with one hidden layer and radial activation functions in hidden layer, are called Radial Basis Function (RBF) networks. Park and Sandberg showed that RBF neural networks are capable of universal approximation[25][26]. Krytzek et al proved the universal convergence of RBF neural networks in function estimation and classification [27]. The overall input-output transfer function of RBF networks is defined as,

$$y_j = \sum_{i=1}^{N_1} w_{ji} \cdot \sigma(|x - \theta_i|), j = 1, \dots, N_y \quad (2.8)$$

where θ_i is the center of the radial basis function of i^{th} hidden neuron, w_{ji} is the link from i^{th} hidden neuron to the j^{th} output neuron.

Two of the most popular RBF activation functions are as follows.

$$\sigma(t) = \exp \left(-\left(\frac{t}{\lambda}\right)^2 \right), \quad (2.9)$$

which is a gaussian function in which λ is a function parameter and

$$\delta(t) = (c^2 + t^2)^\beta, \quad 0 < \beta < 1, \quad (2.10)$$

which is a multiquadratic function with c and β being function parameters. MLP's and RBF's behave differently as a result of the difference in their activation functions in the

following ways:

1. The activation function of each hidden neuron in an MLP neural network computes the inner product of the input vector and the weight vector of that neuron, whereas the activation function of the RBF neural network computes the Euclidean norm between input and vector and center of that neuron.

2. MLP neural networks make global approximations to the nonlinear input-output mapping. In other words, they have generalization capability in the regions where there is not enough data available, but RBF neural networks are capable of quick learning and exhibit reduced sensitivity to the order of training data, since they use a Gaussian function. As a result, a hidden neuron affects the output of a network only for the inputs which are close enough to its center. It is suggested that RBF neural networks work best for the problems with a smaller number of inputs.

2.3.2 Other Neural Network Structures

MLP and RBF belong to those type of structures in which no problem-dependent information is embedded, and as a result, all information about the problem comes from training data. In such a case a large amount of training data is needed to ensure model accuracy. Since gathering that much training data may be practically impossible, the knowledge can be embedded in the network if prior knowledge about the problem exists, to address the above mentioned problem.

There are several recent approaches to the use of prior knowledge in developing neural models. The first approach, which is proposed in [12], is a hybrid EM-ANN model. In this method, the existing equivalent models are used to construct the input-output rela-

relationship of the problem at hand. Then an EM simulator is used to develop a neural model to correct the error between the equivalent model and the actual model. This method is called the difference method. The second method in this approach is called the prior knowledge input method (PKI) [13]. In this method, the source model outputs are entered into the neural network model in addition to the original problem inputs. As a result, the relationship which must be learned is that between the output response of the source model and that of the target model. In [13] it was shown that the performance of the PKI is better than the difference method for a two port GaAs microstrip ground Via. The second approach directly embeds the prior knowledge into neural network structure [28]. This approach is implemented by two methods. The first method uses the symbolic knowledge in the form of the rules to establish the structure and weights in a neural model [28] [29] [30]. The weights or both the weights and topology, can be updated during training [31][32]. In the second method of this approach, the prior knowledge is used to build a modular neural network structure. During this process, the number of necessary modules and the way they interact with each other are defined [33] [34].

A directly microwave oriented Knowledge Based Neural Network (KBNN) was proposed in [1]. This network belongs to the second approach. In this method, the knowledge is incorporated in the neural network in the form of analytical approximation or empirical functions. Since each empirical model is valid only in a certain region of parameter space, a neural model for the entire space can be obtained by providing all the empirical formulae and the mechanism to switch between them. This switching mechanism expands the feature of the Sigmoid Radial Basis Function [35] into high dimensional space and with more generalized activation functions. In this model, like all neural net-

work models, the scale and position of each knowledge function and exact location of each switching boundary is randomly initialized and finally determined during the training process.

As another feedforward neural network structure, Wavelet networks have recently been used in microwave modeling. Although wavelet theory has offered efficient algorithms for different purposes, their implementation is usually limited to the wavelets of small size. The idea of combining the wavelet theory with neural networks has been recently proposed [36] [37] [38]. Since neural networks are powerful tools to solve various problems of large size, the combination of wavelets and neural networks can hopefully remedy the inherent weakness of each structure, resulting in networks which are capable of solving problems of moderately large dimensions. The wavelet networks are feedforward networks with one hidden layer and use wavelets as hidden neuron activation functions. The wavelet network with radial wavelet functions can be considered as an RBF network, since both have the localized basis function. The only difference between them is that the wavelet function is localized in input and frequency domains [38]. This network has a guaranteed upper bound on the accuracy of approximation [39] with a multiscale structure, and the advantage of faster training. Wavelet networks have been used in non parametric regression estimation [36] based on noisy observation to avoid the undesirable local minima [39].

Besides the wavelet network, there are a number of constructive networks, the most representative among them being the cascade correlation network (CasCor) [40]. This network begins with minimal network (without hidden neurons) and then automatically adds hidden neurons one by one during training. The newly added hidden neuron is

then linked to all the neurons. For regression tasks, a CasPer algorithm [41] which constructs a neural network structure in a similar way as CasCor, tends to produce hidden neurons that saturate, thus making CasCor more suitable for classification rather than regression purposes. But both the CasCor and CasPer networks may result in high fan-in to hidden neurons. A towered cascade network is proposed in [42] to overcome this problem.

Another constructive neural network structure, Projection Pursuit Learning Network (PPLN), which adds neurons one by one in a single hidden layer without cascaded connections [43]. The CasCor learns the higher order features using cascaded connection, while PPLN learns it using the trainable activation functions. Whenever a new hidden neuron is added to PPLN, it first trains the new neuron by cyclically updating the output layer weights and activation function and input layer weights associated with the neuron. Then a backfitting procedure is used to fine-tune the parameters of the existing hidden neurons.

2.4 Neural Network Training

Training is an important part of the model development, since an untrained model can't represent input/output relationships. Training can be viewed as an optimization problem with the objective of reducing the error between predicted response by the model and the simulated data. The choice of training algorithms is also one of the key factors in convergence and accuracy of the model. Different problems require different training algorithms depending on the nature of the problem. In this section, a review of training algorithms will be provided with a strong emphasis on second order algorithms.

2.4.1 Objective

The parameters of a neural network should be adjusted so that the neural model can predict the response as desired [15]. These parameters are adjusted through a process called training. The objective of training is to find the parameters such that the error between the neural network predictions and the desired value are minimized, i.e.,

$$\min_{\mathbf{w}} E(\mathbf{w}), \quad (2.11)$$

where $E(\mathbf{w})$ is defined as follows

$$E(\mathbf{w}) = \frac{1}{2} \cdot \sum_{p=1}^{N_p} \sum_{k=1}^{N_y} (y_{pk}(\mathbf{x}_p, \mathbf{w}) - d_{pk})^2 = \frac{1}{2} \cdot \sum_{p=1}^{N_p} e_p(\mathbf{w}), \quad (2.12)$$

where $\mathbf{x}_p$ is the p^{th} input vector and d_{pk} is the k^{th} element of the vector $\mathbf{d}_p$, the desired output vector and $y_{pk}(\mathbf{x}_p, \mathbf{w})$ is the k^{th} output of the neural network when $\mathbf{x}_p$, is presented

to it. $e_p(\mathbf{w})$ represents the p^{th} element in the error vector, which is the squared error between desired output and neural predicted output for the p^{th} sample. To find $\mathbf{w}$, iterative algorithms are used to explore the parameter space, since $E(\mathbf{w})$ is nonlinear with respect to $\mathbf{w}$ and is complicated. An initial guess for $\mathbf{w}$ is made and updated with each iteration. The updated weight $\mathbf{w}_{new}$ can be found using the previous weight, $\mathbf{w}_{old}$ and $\mathbf{d}$, the direction vector, as

$$\mathbf{w}_{new} = \mathbf{w}_{old} + \eta \bullet \mathbf{d}, \quad (2.13)$$

where η is a positive step size defining the extent to which the weights can be changed in that direction.

Note that the way the weights are updated is different for each training algorithm.

2.4.2 Different Types of Training Algorithms

In this section, training algorithms such as backpropagation and second order algorithms are discussed. These algorithms are widely used in neural network training and as such, these algorithms are discussed in details. More emphasis is placed on second order algorithms, since they are the methods used to train the models for the sake of this work.

(A) Backpropagation Algorithms

The backpropagation technique (BP) was proposed by Rumelhart, Hinton and wil-

liams in 1986 [44]. This algorithm is based on the steepest descent principle, where the weights of neural networks are updated along the negative gradient direction in weight space according to the formulae below

$$\Delta w_{old} = w_{new} - w_{old} = -\eta \cdot \frac{\partial}{\partial w} E(w) \Big|_{w = w_{old}}, \quad (2.14)$$

or

$$\Delta w_{old} = w_{new} - w_{old} = -\eta \cdot \frac{\partial}{\partial w} e_p(w) \Big|_{w = w_{old}}, \quad (2.15)$$

where η is learning rate and defines the weight update. Updating can be done in two ways, sample by sample or the batch mode. For batch mode training, where the weights are updated after all training samples have been presented to the network, the training is done according to equation (2.14) and for sample by sample training, where the weights are updated after each sample has been presented to the network, (2.15) is used. This technique introduces convergence problem and possible weight oscillation. To improve the method, a momentum term was added to the update formula, which eliminated the weight oscillation problem. The complete formulae for weight update are presented below for both batch mode and sample by sample training respectively,

$$\Delta w_{new} = -\eta \cdot \frac{\partial}{\partial w} E(w) \Big|_{w = w_{old}} + \alpha \cdot \Delta w_{old} \quad (2.16)$$

or

$$\Delta w_{new} = -\eta \cdot \frac{\partial}{\partial w} e_p(w) \Big|_{w = w_{old}} + \alpha \cdot \Delta w_{old} \quad (2.17)$$

where

$$\Delta w_{old} = w_{new} - w_{old} \quad (2.18)$$

and α is the momentum factor to control the impact of the last weight update on the current update. During the training, the weights are updated at each iteration by a predefined step size in a predetermined direction. For standard backpropagation, the learning rate is adjusted adaptively. This method is very simple and doesn't need the repetitive computation of the error function. Another approach to enhancing the training efficiency is gradient reuse algorithm. In this method, the gradients computed during training are reused until the weight updates lead to a further reduction in training error.

(B) Gradient Based Training Algorithms

The backpropagation algorithm based on steepest descent method can be easily implemented. But the error surfaces encountered in training, sometimes contain planes with a gentle slope. The gradient is too small to cause the weights to move rapidly on error surfaces and consequently reduce the rate of convergence. Also, if the error surface contains a “narrow valley”, the rate of convergence with “the steepest descent” method is very slow, since the direction of the gradient is almost perpendicular to the surface at the “valley”. This causes the oscillation of the update direction and consequently the weight oscillation occurs. To improve the rate of convergence, higher order training methods using gradient information can be used. In this method, the weight w are updated so that

$$E(w_{new}) = E(w_{old} + \eta \cdot d) < E(w_{old}), \quad (2.19)$$

where d is the update direction vector, η is the learning rate, with w_{new} and w_{old} are the current updated weight and previous weight respectively. Different descent algorithms are

distinguished by the method they use to define the successive update directions. Once d is found, the step size can be found by line search method, using $\eta' = \min(\Phi(\eta))|_{\eta > 0}$

where

$$\Phi(\eta) = E(w_{old} + \eta \bullet d) \quad (2.20)$$

The gradient vectors are calculated in the opposite direction to the output flow from each output neuron, exactly the same way in the backpropagation algorithm. For example, for MLP, the chainary rule in derivation is used, starting from the output layer, as shown below

$$\frac{\partial E}{\partial w^i} = \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial \bar{y}^i} \cdot \frac{\partial}{\partial w^i} \bar{y}^i, \text{ where } i = I, \quad (2.21)$$

and then moving through hidden layers to input layers:

$$\frac{\partial E}{\partial w^i} = \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial \bar{y}^j} \cdot \dots \cdot \frac{\partial \bar{y}^{i+1}}{\partial \bar{y}^j} \cdot \frac{\partial \bar{y}^j}{\partial w^i} \quad (2.22)$$

where $i = I - 1, I - 2, \dots, 1$, y is the outputs of the neural network and $\bar{y}^i$ is the outputs from the i^{th} hidden layer.

(B.1) Conjugate Gradient Algorithm [45]

In this method, an initial gradient is obtained from the initial guess of

$w = w_{initial}$, according to $g_{initial} = \left. \frac{\partial E}{\partial w} \right|_{w = w_{initial}}$, and the direction vector

$d_{initial} = -g_{initial}$, is used to recursively find the vectors d, g, γ, λ according to the fol-

lowing formulae:

$$\mathbf{g}_{new} = \mathbf{g}_{old} + \lambda_{old} \cdot \mathbf{H} \cdot \mathbf{d}_{old}, \quad (2.23)$$

$$\mathbf{d}_{new} = -\mathbf{g}_{new} + \gamma_{old} \cdot \mathbf{d}_{old}, \quad (2.24)$$

$$\lambda_{old} = \frac{\mathbf{g}_{old}^T \cdot \mathbf{g}_{old}}{\mathbf{d}_{old}^T \cdot \mathbf{H} \cdot \mathbf{d}_{old}}, \quad (2.25)$$

$$\gamma_{old} = \frac{\mathbf{g}_{new}^T \cdot \mathbf{g}_{new}}{\mathbf{g}_{old}^T \cdot \mathbf{g}_{old}}, \quad (2.26)$$

(Fletcher-Reeves) [46]

or

$$\gamma_{old} = \frac{(\mathbf{g}_{new} - \mathbf{g}_{old})^T \cdot \mathbf{g}_{new}}{\mathbf{g}_{old}^T \cdot \mathbf{g}_{old}}. \quad (2.27)$$

(Polack-Ribiere) [47]

Instead of using the Hessian matrix $\mathbf{H}$ to compute $\mathbf{g}_{new}$ which is computationally extensive, the algorithm proceeds from $\mathbf{w}_{old}$, towards the local minimum $\mathbf{w}_{new}$, with line mini-

mization, and set $\mathbf{g}_{new} = \left. \frac{\partial E}{\partial \mathbf{w}} \right|_{\mathbf{w} = \mathbf{w}_{new}}$. This way the Hessian matrix is not needed in

calculating $\mathbf{g}_{new}$ and, subsequently, we don't have to compute λ_{old} as well. One of the advantages of using this method is that it is very efficient since the descent direction is always along the conjugate direction, which is calculated without getting involved in matrix computations. A disadvantage with this method is that the convergence properties

are not assured *a priori* and depend on the degree to which a local quadratic approximation can be applied to the training error surface. That is because the error surface is not quadratic with respect to w , for neural networks. To overcome this problem, an inexact line search was proposed and a new definition of search direction was given. In order to decrease the computational complexities, a scaled conjugate gradient (SCG) algorithm [48] was introduced to avoid the necessity for line search for each learning iteration.

(B.2) Quasi-Newton Algorithm

The Quasi-newton algorithm [49] was derived from the quadratic objective function. The weights are updated using the formula $w_{new} = w_{old} - \eta \cdot B_{old} \cdot g_{old}$ where B is the inverse of the Hessian matrix, i.e.,

$$B = H^{-1}. \quad (2.28)$$

In this method, we never calculate B directly from the Hessian matrix, instead it is estimated using rank 1 or rank 2 updates, following each line search in a sequence of search directions. There are two major rank two formulae to estimate B_{new} , which are

$$\Delta B_{new} = \frac{d \cdot d^T}{d^T \cdot \Delta g} - \frac{B_{old} \cdot \Delta g \cdot \Delta g^T \cdot B_{old}}{\Delta g^T \cdot B_{old} \cdot \Delta g}$$

(2.29)

Or,
and

$$\Delta B_{new} = \left(1 + \frac{\Delta g^T \cdot B_{old} \cdot \Delta g}{d^T \cdot \Delta g} \right) \left(\frac{d \cdot d^T}{d^T \cdot \Delta g} \right) - \frac{d \cdot \Delta g^T \cdot B_{old} + B_{old} \cdot \Delta g \cdot d^T}{d^T \cdot \Delta g} \quad (2.30)$$

where

$$\Delta B_{new} = B_{new} - B_{old}, d = w_{new} - w_{old}, \Delta g = g_{new} - g_{old} \quad (2.31)$$

The first formula is called the DFP (Davidson-Fletcher-Powell) formula and the second one is called the BFGS (Broyden-Fletcher-Goldfarb-Shanno) formula. Another issue worth mentioning is the storage space required to maintain an estimation of Hessian matrix in a standard Quasi-Newton method. If the number of the weights is N_w , then N_w^2 space is necessary to store the matrix. A limited memory (LM) or BFGS were applied in which the inverse Hessian matrix is set to identity matrix after each iteration. By estimating the inverse Hessian matrix, the Quasi-Newton method has faster convergence than the conjugate gradient method. The theoretical convergence rate and practical performance of the second-order gradient-based methods generally outperform the first order method.

(C) Other Training Methods

One of the important least square methods is the Gauss-Newton algorithm. This method can be also used to estimate the neural model parameters. It is a linearization method. Let v be a vector containing the individual error terms in (2.12) and J be the

$(N_p \times N_y) \times N_w$ Jacobian matrix which contains the derivative of v w.r.t the weights w .

The update formula for the Gauss-Newton model can be expressed as

$$w_{new} = w_{old} - ((J_{old}^T \cdot J_{old})^{-1} \cdot J_{old}^T \cdot v_{old}). \quad (2.32)$$

In formula (2.32), $J_{old}^T \cdot J_{old}$ is positive definite unless J_{old} is rank deficient. The Levenberg-Marquardt method [50] can be applied if J_{old} is rank deficient and in this case the update formula is given by (2.33)

$$w_{new} = w_{old} - ((J_{old}^T \cdot J_{old} + \mu \cdot I)^{-1} \cdot J_{old}^T \cdot v_{old}), \quad (2.33)$$

where $\mu \geq 0$.

By using a diagonal matrix, instead of the identity matrix I in (2.33), the training of the multilayer feedforward network can be done more efficiently, as is proposed in [51]. To further improve the efficiency of the algorithm, the size of the Jacobian matrix is reduced by dividing the training samples into several groups called local batches, as is proposed in [52]. The training is successively performed through these local batches.

As is seen, using the second order training algorithms has resulted in a much more efficient training, for example they are much faster than the backpropagation method. However, they require the storage of the inverse Hessian matrix and its estimation. Training a large neural network can be a large scale optimization. To solve the problem of large scale optimization, the decomposition method has been developed and several training methods are proposed in [53] [54] [55] to decompose the training task by training the neu-

ral network layer by layer. Linear programming can be used to solve the large scale optimization problems, for example in [56], the neural network training was linearized and formulated as a constrained linear programming. A combination of linear and nonlinear programming techniques could reduce the degree of nonlinearity of the error function w.r.t the hidden layer weights and therefore reduce the chances of being trapped in a local minima. Therefore, a hybrid training algorithm is highly necessary for a large scale problem [57]. In the case of RBF networks, they are trained by a decomposed process. The nonlinear hidden layers can be regarded as *centers* and *widths*, to make it possible to organize the neurons in such a way that they reflect the distribution of the training data. By doing so, the hidden layer weights can be set through unsupervised training and output layer weights can be obtained through linear LMS algorithm. On the other hand, it is not easy to develop the heuristics to initially assign the weights for the hidden layers in MLP, because of its black box approach. But by using the hybrid linear and nonlinear method, proposed in [57], the best main characteristics of the linear LMS algorithm of RBF and nonlinear optimization techniques are integrated together in one routine. This routine can be made suitable for use in any feedforward neural network with linear output weights. With this method, the number of independent parameters are reduced and a guaranteed global minimum with respect to output layer weight can be reached.

Another important category of algorithm is the global training algorithm. This technique contains a random search element in its training process to allow the training process to escape from local minima and possibly converge to a global minimum of objective function. The important methods in this class are: simulated annealing which allows the optimization to move from a local minimum through a process called annealing con-

trolled by temperature [58]; genetic algorithm which is inspired from the biological evolution, changes the structure of the weights and weights of a neural network through several generations [59]; the Langevin updating (LV) technique in MLP, where noise is added to the weights during training [60].

Chapter 3

Library of Neural Models for Microwave Components

In this chapter, a brief discussion of the library development will be provided. Then the developed neural models for the components will be discussed along with their corresponding results, including testing error. Finally, the developed models will be presented. Multilayer perceptrons (MLP) are utilized to learn the input/output relationship and to generalize it. The testing samples presented to the models developed are independent from the training samples. The small error between model outputs and desired outputs shows the ability of the model to learn and to generalize consequently. The efficiency of the model is demonstrated. Later these models are used in repetitive analysis, such as yield analysis or optimization (by the Monte-Carlo method), where these models prove their superiority to their EM counterparts.

3.1 Library Development: Problem Statement and Steps

In this section, the objective is to develop a library of microwave passive components. As a matter of fact, the realistic power of many CAD tools depends on the richness, speed and the accuracy of their library models. If the library consists of, N_m component models, and the model of interest in the library is the n_{th} model in the library for which the input and output are represented by X^n and Y^n accordingly, the library development can be defined as creating models to represent the multidimensional nonlinear relationship $Y^n = Y^n(X^n)$ for every n where $1 \leq n \leq N_m$. The sets of the values represented by X^n and Y^n are called X^n space and Y^n space respectively. For example, to model a bend, for design purposes, Y^n can represent “S” parameters such as s_{11} , s_{12} , s_{21} , s_{22} , and X^n can represent the physical/geometrical parameters, such as, frequency f , width of the conductor and height of the substrate. To develop a library, training data should be obtained for different value combinations in X^n space, namely all the input parameters and Y^n space, namely outputs of interest. It is worth mentioning that the accuracy of the library depends on the accuracy of the training data, so every measure should be taken to prepare the training data as accurately as is required for the library. For the sake of this work, the EM simulations were used as the most accurate way to obtain the training data. However, obtaining highly accurate data is very time consuming, so there is always a trade-off

between the needed accuracy and the costs. For the sake of this work, hours of time have been spent by the EM simulator to obtain highly accurate data. Bearing in mind that in the real world this simulator is used repeatedly during the design cycle and the fact that each simulation takes a long time to complete, the developed library is capable of saving on the design time during the design cycle to address the time-to-market challenge due to its fast response after training. In summary the following steps have been taken to develop the library:

1. Define input and output spaces of each model in the library.
2. Collect training data for various value combinations of input parameters of each model through HP Momentum simulation to obtain output responses. Note that training data should be sufficient to obtain reliable models.
3. The MLP neural structure is used and trained by backpropagation, conjugate gradient and Quasi-Newton methods as discussed in earlier sections.
4. If the testing error was not satisfactory, the number of hidden neurons was increased and if no noticeable improvement was made, the training samples were increased and the model was trained using the new set of training samples to obtain a more reliable model until a satisfactory model accuracy is achieved.

In this chapter, a library of basic microwave passive microwave components is identified which includes single and coupled microstrip line, 90 degree bend and variable angle bend, T-Junction, open and short circuit stub, radial and butterfly stub and microstrip width taper. Then a library of neural models has been developed for the corresponding microwave components in the library. Common parameters for these components are as

follows (except the single transmission line):

$$\epsilon_r = 12.9, TH_{sub} = 4mil, \tan\delta = 0.0002, \quad (3.1)$$

where $\epsilon_r = 12.9$ is relative dielectric constant, $TH_{sub} = 4mil$ is substrate thickness and $\tan\delta$ is the loss tangent of the substrate. Metal conductance is $2.7 \times 10^7 S/m$ and metal thickness is about 1 mil and the port impedance is 50Ω . S parameters are obtained relative to this impedance. These values are constant for all the models unless otherwise mentioned. Note that the worst case error for all models doesn't exceed 1%.

For the single transmission line only the loss tangent is a constant and the other parameters are variables. For other models the above mentioned parameters are fixed. The parameter descriptions of the entire library are listed in Table 3.1. The inputs to each model and their corresponding ranges are shown in their appropriate tables for each component as will be noted in each section.

Parameter	Explanation
<i>Frequency</i>	frequency
<i>L</i>	Length of microstrip lines
ϵ_r	Relative dielectric constant
TH_{sub}	Substrate thickness
<i>S</i>	Microstrip line spacing
<i>ANG</i>	Angle of bends
<i>Mitter Fraction</i>	Mitter fraction (Only for variable angle bends)
<i>RO</i>	Outer radius for butterfly stub

Table 3.1: Parameter description for the entire library.

Parameter	Explanation
W	Width of a single or coupled microstrip line or a stub
w_1	One of the 3 widths of a T-Junction or 2 widths of a Width taper
w_2	One of the 3 widths of a T-Junction or 2 widths of a Width taper
w_3	One of the 3 widths of a T-Junction
w_{in}	Input width of a symmetric T-Junction
w_{out}	Output width of a symmetric T-Junction

Table 3.1: Parameter description for the entire library.

3.2 Microwave Component Neural Models

The microwave neural models are developed and the accuracy of the models are presented in this section. The testing error of the obtained models is a good judge of the accuracy and generalization capability of the neural model developed. Different neural models have been obtained for each component by training with different training sample sizes. The best model obtained for each training sample size is highlighted in the appropriate table.

3.2.1 Transmission Line Neural Models

Neural network models were developed for single and coupled microstrip lines. Transmission line models are essential for the analysis of delay and cross talk in high-speed VLSI interconnect design. EM simulations of transmission line responses are very

slow and time consuming, especially if they need to be repetitively done. Neural models trained off-line from EM data, can be used on-line during VLSI interconnect design providing fast solutions to the original problem. The training and testing data were obtained by simulation of single microstrip line and double microstrip line using HP Momentum analysis based on the method of moments as explained in [7] and trained by NeuroModeller [8].

The neural model for single microstrip line is shown in Figure 3.1. The input parameters of the single microstrip line neural model are: frequency (f), width (w) and length (L) of conductor, substrate thickness (TH_{sub}) and dielectric constant (ϵ_r). These parameters are shown more clarified in Figure 3.2 which shows the geometry of a single microstrip line. The definitions of these parameters are given in Table 3.1 and their appropriate ranges are shown in Table 3.2. The output parameters of the model are magnitude and phase of s_{11} and s_{12} since the geometry is symmetric since other S outputs aren't necessary. Five sets of data were generated. It took about 15 hours to generate the highly accurate data. The resulting training and testing errors for all different sample sizes and number of hidden neurons are shown in Table 3.3 for single microstrip line. The MLP structures in this work are denoted as N_x - N_z - N_y where N_x is the number of input neurons, N_z is the number of hidden neurons and N_y is the number of output neurons. As is evident, increasing the number of hidden neurons decreases the testing error until a certain point beyond which no improvement can be made. At this point, the training sample size is increased to decrease the testing error. Different models were trained for single microstrip line with different training sample size. The best model obtained, in terms of lower testing error, for each training sample size is highlighted. Figure 3.3, shows that the higher the

number of training samples is, the lower the testing error becomes. Note that the testing data set was never used in the training which ensures that the reduced testing errors can only be attributed to the learning and generalization capability of the developed neural model. As seen from Table 3.3, the errors cannot be significantly reduced once the number of training samples exceed a certain limit and no improvement can be reached in the model. In addition, for each specific sample size, the testing error increases by increasing the number of hidden neurons since the neural model can't learn the relationship and simply memorizes the samples. As Figure 3.3 suggests, the best model for single microstrip line is 5-63-4 since the testing error can't be lowered any further. The right training sample size is 1100, since the testing error can't be decreased noticeably any further.

Single microstrip line	<i>Frequency (GHz)</i>	<i>w (mil)</i>	<i>L (mil)</i>	ϵ_r	TH_{sub} (mil)
	$0.5 < f < 55$	$0.8 < w < 2.8$	$50 < L < 200$	$2.32 < \epsilon_r < 12.9$	$4 < TH_{sub} < 10$

Table 3.2: Inputs to single microstrip line neural model.

Figure 3.1 Neural network structure for a single microstrip line.

Figure 3.2 A single microstrip line (a) 3 dimensional view of a single line microstrip line and (b) 2 dimensional view of a single microstrip line along with its geometrical parameters. Substrate relative dielectric constant is ϵ_r .

No of training samples	Neural Model Structure	Average Test Error
150	5-10-4	4.36%
	5-15-4	3.11%
	5-16-4	2.49%
	5-18-4	2.52%
300	5-20-4	3.71%
	5-30-4	2.45%
	5-35-4	1.46%
	5-40-4	1.51%
600	5-30-4	1.76%
	5-35-4	1.23%
	5-38-4	.76%
	5-40-4	.81%
900	5-30-4	3.19%
	5-50-4	.94%
	5-55-4	.71%
	5-60-4	.78%
1100	5-50-4	1.3%
	5-60-4	.83%
	5-63-4	.69%
	5-70-4	.72%

Table 3.3: All trained neural models for single microstrip line are tested with one set of testing data. The tendency shows that by increasing the hidden neurons the testing error decreases until the right number of training sample beyond which no further improvement in testing error can be reached.

Figure 3.3 Model accuracy comparison between different models trained by different sizes for single microstrip lines. Testing data is in the same range as the training data. Here the best models trained by each training data samples are chosen and compared against each other. Note that the horizontal axis is number of samples and vertical axis is average testing error.

The responses of the single microstrip line, namely, magnitude and phase of S_{11} and S_{12} are displayed in Figure 3.4 which gives a comparison of the neural predicted outputs and the EM simulator accurate outputs for testing data. The two responses are quite similar which shows the accuracy of the predicted neural response. Figure 3.5 shows the predicted relationship of the responses versus input parameters from neural model for single microstrip line neural models.

Figure 3.4 Comparison between the simulator response and neural model response for (a) $|S_{11}|$ (b) $|S_{12}|$ (c) $\angle S_{11}$ and (d) $\angle S_{12}$ of the single microstrip line. As is evident, the responses are almost identical which shows that the model is accurate. The vertical axis is testing error (percentage) and the horizontal axis is the sample number.

Figure 3.5 The frequency response for (a) $|S_{11}|$ (b) $|S_{12}|$ (c) $\angle S_{11}$ (d) $\angle S_{12}$ for single microstrip line with horizontal axis being the frequency and vertical axis being the S parameter of interest. S parameters are measured with the terminating resistor of 50 ohms.

Figure 3.6 Neural network structure for a coupled microstrip line.

coupled microstrip line	Frequency (GHZ)	W (mil)	L (mil)	S (mil)
	$0.5 < f < 55$	$0.8 < w < 2.8$	$50 < L < 150$	$1 < S < 1.7$

Table 3.4: Inputs to the coupled microstrip line model.

The inputs to the microstrip coupled line model are (f)-frequency, (L)-length of the

lines, (S)-spacing between the lines and (w)-the width of the lines as given in Table 3.4 along with their appropriate ranges. The neural model for a coupled microstrip line is shown in Figure 3.6. These parameters are shown with more clarity in Figure 3.7, the geometry of the coupled microstrip line.

The outputs of the model are real and imaginary parts of S parameters namely S_{ijR} and S_{ijI} where the suffixes R and I denote the real and imaginary part of S_{ij} respectively, and i, j denote the ports at which the S parameters are being simulated. Six sets of data were generated to obtain the best trained model for this component. The results are shown in Table 3.5. Figure 3.8 shows the testing error of the neural model for different outputs and Figure 3.9 shows the frequency response of the model developed for the coupled microstrip line corresponding to different lengths. For both single and coupled lines, the Input/Output relationship for the components is modeled by the neural network developed since a good agreement can be observed.

The advantage of the neural network approach over the accurate EM simulation becomes evident when the total time for each EM simulation by the method of moments is compared to the neural network instant response. Each EM simulation takes about 1 hour to complete especially since the number of meshes are high and the larger the number of meshes, the longer each simulation takes. The neural network can perform each simulation in 20 seconds. The training took about 3 and 4 hours for single and coupled microstrip lines respectively with a relatively low testing error. The second order methods are known to have a faster convergence. However, if they are used in the first stage of training, they do not show a strong performance. As such, the model was first trained with the backpropagation method followed by the conjugate gradient method. The Quasi-Newton method

was used in the last stages of training because of its fast convergence when reaching the minimum of the error surface.

Figure 3.7 The geometric view of a coupled microstrip line (a) 3 dimensional view (b) 2 dimensional view

No of training samples	structure	test error
500	5-50-8	8.36%
	5-70-8	4.21%
	5-80-8	3.31%
	5-92-8	3.47%
1000	5-60-8	2.23%
	5-70-8	1.82%
	5-77-8	1.51%
	5-80-8	1.42%
1300	5-60-8	2.12%
	5-80-8	1.31%
	5-82-8	1.25%
	5-85-8	1.23%
1700	5-70-8	2.17%
	5-80-8	1.02%
	5-83-8	.94%
	5-90-8	.95%
2000	5-60-8	2.20%
	5-90-8	.97%
	5-95-8	.93%
	5-98-8	.96

Table 3.5: All trained neural models for coupled microstrip line are tested with one testing data. The tendency shows that by increasing the hidden neurons the testing error decreases until the right number of training sample beyond which no further improvement in testing error can be reached.

Figure 3.8 Testing errors (vertical axis) for different samples (horizontal axis).

Figure 3.8 Testing errors (vertical axis) for different samples (horizontal axis).

Figure 3.9 The neural model frequency response obtained for a coupled microstrip line for (a) s_{11R} , (b) s_{11I} , (c) s_{12R} , (d) s_{12I} , (e) s_{13R} , (f) s_{13I} , (g) s_{23R} , (h) s_{23I} with horizontal axis being the frequency and vertical axis being the relevant "S" parameter. The length is variable.

Figure 3.9 The neural model frequency response obtained for a coupled microstrip line for (a) S_{11R} , (b) S_{11I} , (c) S_{12R} , (d) S_{12I} , (e) S_{13R} , (f) S_{13I} , (g) S_{23R} , (h) S_{23I} with horizontal axis being the frequency and vertical axis being the relevant "S" parameter. The length is variable.

3.2.2 Microstrip Bend Neural Models

Neural models for bend structure, namely a 90 degree bend and an arbitrary angle bend were developed. The geometry for a 90 degree bend is shown in Figure 3.10 and its corresponding neural model structure is shown in Figure 3.11. The model inputs are frequency (f) and width (W) whose definitions and ranges are in Table 3.1 and Table 3.6. The model outputs are S parameters- S_{11R} , S_{11I} , S_{12R} , S_{12I} , S_{22R} , S_{22I} for 90 degree bend.

Four sets of data were generated for the 90 degree bend. Table 3.7 shows that by increasing the number of hidden neurons can improve the testing error until a certain point and to further improve the testing error, the training sample size should be increased. The training data sets were generated by simulating the original problem using HP Momentum analysis to provide highly accurate samples. The numbers of testing samples are 800 for 90 degree bend and all different structures for this component are tested with one unique testing data set which has never been used during the training to ensure the reliability of the model through the testing errors. It seems that 2-45-4 trained by 800 training samples is the best model for the 90 degree bend.

Figure 3.10 Geometrical view of a 90 degree bend (a) 2 dimensional (b) 3 dimensional.

Figure 3.11 Neural network structure for 90 degree bend.

Name Of The Component	Frequency (GHZ)	W (mil)
90 degree bend	$0.5 < f < 55$	$1 < w < 4.8$

Table 3.6: Inputs to the neural 90 degree bend and their ranges.

No of training samples	structure	test error
150	2-10-4	5.47%
	2-15-4	3.23%
	2-20-4	2.78%
	2-24-4	2.92%
400	2-15-4	3.15%
	2-20-4	1.46%
	2-28-4	1.08%
	2-32-4	1.12%
600	2-20-4	1.63%
	2-25-4	1.09%
	2-35-4	.88%
	2-42-4	.91%
800	2-25-4	1.14%
	2-35-4	.95%
	2-45-4	.87%
	2-48-4	.89%

Table 3.7: All trained neural models for microstrip bend are tested with one testing data set. The tendency shows that by increasing the hidden neurons the testing error decreases until the right number of training sample beyond which no further improvement in testing error can be reached.

Figure 3.12 The comparison between predicted and EM simulated results for a microstrip bend. The horizontal axis is sample number and the vertical axis is relative S parameter.

Figure 3.13 Frequency response of the microstrip bend model. The input width is being varied. The horizontal axis is frequency and the vertical axis is the relevant S parameter.

Figure 3.14 (a) 2 dimensional (b) 3 dimensional view of an arbitrary angle bend.

By careful examinations of Figure 3.12 (the comparison between output of the neural model and EM simulator) the neural model is found to be almost as reliable as the accurate EM/physical model. Figure 3.13 shows the neural model frequency response for different widths. The geometry for a variable angle bend is shown in Figure 3.14 and its neural model is shown in Figure 3.15. The inputs to the model are w the width and Ang , the variable angle and the frequency as shown in Table 3.8. The model outputs are S parameters-, $|S_{11}|$, $\angle S_{11}$, $|S_{21}|$, $\angle S_{21}$, $|S_{22}|$, $\angle S_{22}$. Different models were trained using different training sample sizes to find the best possible model. The results are shown in Table 3.9 all of which are tested with one unique testing data. It seems that 3-90-4 trained by 1800 data samples is the best model obtained for variable angle bend. As is evident from Figure 3.16, testing error is low which is the indication that the neural model is quite accurate. The neural model frequency responses corresponding to variations in width are shown in Figure 3.17.

The training took about 2 hours to complete, and once it is completed, the model performs the simulation in 20 seconds which is an obvious advantage to compare to EM simulation since an accurate EM analysis takes several hours to complete. Once the model has been developed, it can provide the user with instant response to different problems and, therefore, can be used on-line repetitively. EM simulations take a considerable time especially when the number of the meshes is increased. In this case, the simulations take about 1 hour to complete. Figure 3.18 (a), (b) show the error tendency for different numbers of samples for the 90 degree bend and the variable angle bend respectively. These models were trained by successive applications of backpropagation, conjugate gradient and Quasi-Newton training algorithms, since in this method, the convergence occurs rap-

idly due to the fast convergence of the Quasi-Newton method when it is close to the minimum and subsequently the training occurs faster. Another strategy is training the model by the first application of conjugate gradient and then Quasi-Newton training algorithms which takes longer than the previous method.

Figure 3.15 Neural network structure for arbitrary angle bend.

Name Of The Component	Frequency (GHZ)	W (mil)	Ang (Deg)
Arbitrary bend	$0.5 < f < 55$	$1 < w < 3.6$	$10 < ANG < 90$

Table 3.8: Inputs to the arbitrary angle model

No of training samples	structure	test error
340	3-50-4	5.43%
	3-75-4	4.43%
	3-85-4	3.47%
	3-90-4	3.52%
700	3-50-4	4.24%
	3-65-4	2.56%
	3-75-4	1.83%
	3-75-4	1.87%
900	3-60-4	2.87%
	3-70-4	2.15%
	3-80-4	1.46%
	3-83-4	1.57%
1400	3-60-4	2.44%
	3-70-4	1.96%
	3-80-4	.87%
	3-85-4	.92%
1800	3-75-4	1.34%
	3-90-4	.86%
	3-95-4	1.07%

Table 3.9: All trained neural models for arbitrary bend microstrip line are tested with one testing data. The tendency shows that by increasing the hidden neurons the testing error decreases until the right number of training sample beyond which no further improvement in testing error can be reached

Figure 3.16 horizontal axis is the sample number and vertical axis is the relevant S parameter testing error. The predicted results matches that of the testing data obtained from the Simulator within an acceptable range, showing that, the predicted data is quite accurate.

Figure 3.17 Frequency response for different outputs of a variable bend junction with variable width with horizontal axis being the frequency and vertical axis is the relevant S parameter.

Figure 3.17 Frequency response for different outputs of a variable bend junction with variable width with horizontal axis being the frequency and vertical axis is the relevant S parameter.

Figure 3.18 Average percentage testing error versus training samples for a) 90 degree bend and b) variable angle bend. The horizontal axis is the number of samples and the vertical axis is the testing error.

3.2.3 Microstrip Symmetric T-Junction Neural Model

The T-Junction geometries are shown in Figure 3.19 and its neural model is shown in Figure 3.20. This component is used to split the signal into two different paths. In a symmetric T-Junction $w_1 = w_2 = w_{out}$ and $w_3 = w_{in}$. The input parameters and their Minimum and Maximum values are shown in Table 3.10. ϵ_r and TH_{sub} are fixed, as is mentioned in (3.1). The input parameters to neural model are frequency (f), input width (w_{in}) and output width (w_{out}) as is mentioned in Table 3.10. The outputs are S parameters - $S_{11R}, S_{11I}, S_{12R}, S_{12I}, S_{13R}, S_{13I}, S_{22R}, S_{22I}$. Four sets of training data and one set of testing data were generated for T-Junction problem using HP Momentum analysis. Table 3.11 lists the models trained with different training sample sizes. The number of testing samples is 1500. Here the best model seems to be 3-90-8 trained by 1800 samples. Table 3.11 shows that the number of hidden neurons was increased for each training sample size until no further improvement in testing error was observed. The testing error cannot be improved noticeably once the number of training data is increased beyond 1800 samples, which indicates that increasing the number of the training sample size leads to improvement in testing error until a certain point beyond which no significant improvement can be made. The fact that the testing and training models are independent clearly indicates that the neural model has learned the I/O relationship. The scattering plot (The neural readings versus EM simulated results) shown in Figure 3.21 indicates the

accuracy of the model and the I/O relationships learned by the model. Ideally the scattering plot should be the line $y = x$, however since the model prediction is not exactly as accurate as the original physical problem, it is not quite linear. The more accurate the model is, the closer the points are along the line. The frequency response of the model is shown in Figure 3.22. The training took two hours to complete and was accomplished by successive application of the conjugate gradient and Quasi-Newton methods. It is worth mentioning that if the training is performed by first applying the Quasi-Newton method it takes longer to train the model since the convergence is slower at the beginning of the training when the model is trained using this method.

Figure 3.19 (a) two dimensional (b) three dimensional view of T-Junction.

Figure 3.20 Neural network structure for a symmetric T-Junction.

Name Of The Component	Frequency (GHZ)	w_{in} (mil)	w_{out} (mil)
T-Junction	$0.5 < f < 55$	$0.8 < w_{in} < 2.4$	$0.8 < w_{out} < 2.4$

Table 3.10: Inputs to the T-junction neural model

No of training samples	structure	test error
350	3-50-8	5.43%
	3-75-8	5.3%
	3-90-8	4.81%
	3-92-8	4.87%
700	3-50-8	3.7%
	3-65-8	3.2%
	3-75-8	2.12%
	3-83-8	2.23%
900	3-60-8	2.87%
	3-70-8	2.42%
	3-80-8	1.76%
	3-85-8	1.79%
1400	3-60-8	2.12%
	3-70-8	1.46%
	3-80-8	.83%
	3-85-8	.85%
1800	3-75-8	1.45%
	3-80-8	.97%
	3-90-8	.82%
	3-94-8	.85%

Table 3.11: All trained model for microstrip T-Junction (testing data set is almost of the same size as the training data set).

Figure 3.21 Neural predicted response (vertical axis) vs. EM simulated response.

Figure 3.21 Neural predicted response (vertical axis) vs. EM simulated response.

Figure 3.22 The horizontal axis is the frequency and the vertical axis is the relevant S parameter. The output width is being varied.

Figure 3.22 The horizontal axis is the frequency and the vertical axis is the relevant S parameter. The output width is being varied.

3.2.4 Microstrip Open and Short Circuit Stub Neural Models

Open and short circuit stubs are important components for many microwave circuits such as filters and impedance matching networks. The geometry of an open circuit stub is shown in Figure 3.23. The inputs to the model are frequency (f), length (L) and width (W) as is mentioned in Table 3.12 along with their appropriate ranges. S parameters are the output parameters - S_{11R} , S_{11I} - namely real and imaginary part of S_{11} respectively. The training and testing data were generated using HP Momentum analysis. Four sets of training data were generated. The model was trained with different training sample sizes to obtain the best trained model, i.e., with the least testing error. All the neural models obtained were tested with one unique testing data. The results are shown in Table 3.13. The best trained neural model for each training sample size is highlighted. The tendency of these best trained models indicates that beyond a certain point, no further significant improvement can be made in testing error by increasing the training sample size. By examining Figure 3.25 (errors) and Figure 3.26 (neural model frequency response) for different widths, the model developed is found to be almost accurate for open circuited stub and as such, the model has learned the relationship between the Input/Output. The training took about four hours to complete by applying conjugate gradient and Quasi-Newton training algorithms successively.

By increasing the number of meshes, the EM simulator was found to require more time to provide the answer. Taking that factor into consideration, the superiority of the neural model becomes evident since once trained, the response can be immediately found

from the model.

The geometry for a short circuit stub is shown in Figure 3.27 and the neural model is shown in Figure 3.28. The inputs to the model are frequency (f), length (L) and width (W). Table 3.14 shows the inputs and their appropriate ranges. Note that ϵ_r and TH_{sub} are fixed. The outputs of the model are S_{11R} , S_{11I} . Table 3.15 summarizes the results of training the neural model with different sample sizes. The best result for each sample size is highlighted. The tendency of the best models testing errors suggest that the neural structure 3-80-2, trained with 1800 samples is the best model obtained since the error can't be decreased any further by increasing sample size. All the trained models were tested with one set of testing samples. Figure 3.29 shows the error of the neural model. The fact that the neural model is very low suggests that the model has been able to learn the Input/Output relationship. The frequency response of the model is shown in Figure 3.30 corresponding to variation in widths. The training took about six hours to complete by applying conjugate gradient and Quasi-Newton training algorithms successively after the application of backpropagation.

Figure 3.23 Geometry of an open circuited stub.

Figure 3.24 Neural network structure for an open circuited stub.

Name Of The Component	Frequency (GHZ)	W (mil)	L (mil)
Open circuited stub	$5 < f < 55$	$0.8 < w < 2.8$	$50 < L < 150$

Table 3.12: Inputs to the open circuit stub neural model.

No of training samples	structure	test error
300	3-20-2	4.67%
	3-30-2	3.37%
	3-40-2	2.34%
	3-43-2	2.39%
600	3-30-2	2.76%
	3-45-2	1.89%
	3-50-2	1.43%
	3-55-2	1.49%
900	3-40-2	1.93%
	3-50-2	1.23%
	3-60-2	.92%
	3-60-2	.97%
1100	3-40-2	1.67%
	3-50-2	.98%
	3-60-2	.83%
	3-65-2	.86%
1500	3-50-2	1.25%
	3-55-2	1.02%
	3-60-2	.81%
	3-65-2	.84%

Table 3.13: All trained neural models for open circuited stub are tested with one testing data. The tendency shows that by increasing the hidden neurons the testing error decreases until a certain training sample size beyond which no further improvement in testing error can be reached.

Figure 3.25 Errors between neural prediction and actual testing data for an open circuit stub. The horizontal axis is the sample number and the vertical axis is the percentage testing error.

Figure 3.26 Frequency response obtained for output S parameters of an open circuit stub with different widths. The horizontal axis is the frequency and the vertical axis is the relevant S parameter.

Figure 3.27 (a) 2 dimensional (b) 3 dimensional view of short circuit stub.

Figure 3.28 Neural network structure for a short circuit stub.

Name Of The Component	Frequency (GHZ)	W (mil)	L (mil)
short circuited stub	$0.5 < f < 55$	$0.8 < w < 2.8$	$50 < L < 150$

Table 3.14: Inputs to short circuit stub neural model.

No of training samples	structure	test error
400	3-50-2	4.38%
	3-70-2	3.12%
	3-90-2	2.48%
	3-95-2	2.53%
700	3-30-2	3.69%
	3-50-2	2.12%
	3-60-2	1.34%
	3-63-2	1.38%
1000	3-40-2	1.87%
	3-50-2	1.12%
	3-65-2	.85%
	3-70-2	.89%
1400	3-50-2	1.23%
	3-60-2	.97%
	3-70-2	.74%
	3-74-2	.77%
1800	3-50-2	2.01%
	3-70-2	1.15%
	3-80-2	.72%
	3-85-2	.78%

Table 3.15: All trained neural models for short circuited stub are tested with one testing data set. The tendency shows that by increasing the hidden neurons the testing error decreases until the right number of training samples beyond which no further improvement in testing error can be reached

Figure 3.29 Errors between neural prediction and actual testing data for a short circuited stub for a) s_{11R} and b) s_{11I} the figure shows the consistency of the model output and the output from HP Momentum simulator. The horizontal axis is the frequency and the vertical axis is the percent testing error.

Figure 3.30 Frequency response of a short circuited stub with horizontal axis being the frequency. The width is being varied here. The horizontal axis is the frequency and the vertical axis is the relevant S parameter.

3.2.5 Microstrip Radial and Butterfly Stub Neural Models

Radial stubs are widely used in many microstrip circuits such as matching networks, bias lines, low pass filters, etc., where both an accurate localization of a zero point impedance and a very low level of characteristic impedance are needed. The geometry of a radial stub is shown in Figure 3.31. A neural model was developed for the radial stub as shown in Figure 3.32. The inputs to the model are Frequency (f), width (W), length (L) and angle (ANG). Table 3.16 shows the appropriate range of the inputs for radial stub. The outputs of the neural model are S parameters namely S_{11R} and S_{11I} . Training and testing data samples are generated with HP Momentum analysis. Four different training sets were generated. As usual, the trained structure with the best accuracy is selected and highlighted among those trained by each set and are compared together. The tendency of the testing error with respect to number of samples suggests that by increasing the number of training samples the accuracy improves up to a certain point beyond which no noticeable improvement can be observed. It seems that the best model is obtained when the number of training data is 1700. The results shown in Figure 3.33 (scattering plot) suggest a close agreement between the responses from the one predicted by neural model and that obtained from EM simulation. The training has been accomplished by successive application of backpropagation, conjugate gradient and Quasi-Newton training methods. Note that in this problem, without application of Quasi-Newton methods, the convergence is extremely difficult and therefore it takes longer to converge.

The geometry for butterfly stub is shown in Figure 3.35. The neural model is shown in Figure 3.36. The input parameters to this model are the same as that of the radial

stub whose range is shown in Table 3.18. The outputs of the model are S_{11R} , S_{11I} , S_{12R} , S_{12I} . Four different sized training set were generated to train the model. The result of the testing is shown in Table 3.19. Note that these models are all tested with one set of testing samples. The best trained model is highlighted for each training set. By careful examination of the tendency of the errors, the best structure is found to be 4-80-4 trained by 1900 samples.

The errors in Figure 3.37 suggest a close agreement between the accurate EM simulation results with the neural model predictions. The frequency response of the neural model is shown in Figure 3.38 due to the variations in the width. The training has been performed by successive application of backpropagation, conjugate gradient and Quasi-Newton training methods. Superiority of the neural model becomes more clear by noting the fact that the EM simulation takes about 2 hours to complete, however, the neural model takes about 20 seconds. Therefore it is much easier to use it during an on-line design cycle.

Figure 3.31 Radial stub (a) two dimensional b) three dimensional geometrical view

Figure 3.32 Neural network structure for a radial stub.

Name Of The Component	Frequency (GHZ)	W (mil)	L (mil)	ANG (deg)
Radial stub	$0.5 < f < 55$	$0.8 < w < 2.8$	$50 < L < 70$	$10 < ANG < 130$

Table 3.16: Inputs to radial stub neural model.

No of training samples	structure	test error
400	4-30-2	3.32%
	4-40-2	2.97%
	4-50-2	2.45%
	4-56-2	2.48%
900	4-50-2	2.33%
	4-50-2	1.76%
	4-60-2	1.13%
	4-65-2	1.18%
1400	4-55-2	1.23%
	4-70-2	.82%
	4-80-2	.96%
1700	4-75-2	.93%
	4-80-2	.80%
	4-85-2	.81%

Table 3.17: All trained neural models for radial stub are tested with one testing data set. The tendency shows that by increasing the hidden neurons the testing error decreases until the right number of training sample beyond which no further improvement in testing error can be reached.

Figure 3.33 Scattering for radial stub for (a) s_{11R} and (b) s_{11I} . The horizontal axis is the relevant simulated S parameter obtained from HP and the vertical axis is that obtained from neural prediction.

Figure 3.34 Frequency response of a radial stub with horizontal axis being the frequency and the vertical axis is the relevant S parameter. The length is being varied here.

Figure 3.35 A butterfly stub (a) 2 dimensional (b) 3 dimensional view

Figure 3.36 Neural network structure for a butterfly stub.

Name Of The Component	Frequency (GHZ)	W (mil)	L (mil)	ANG (Deg)
Butterfly stub	$0.5 < f < 55$	$0.8 < w < 2.8$	$50 < L < 70$	$50 < ANG < 70$

Table 3.18: Inputs to butterfly stub neural model.

No of training samples	structure	test error
300	4-50-4	5.02%
	4-60-4	4.49%
	4-80-4	3.84%
	4-83-4	3.89%
700	4-40-4	3.7%
	4-50-4	2.9%
	4-60-4	2.54%
	4-65-4	2.63%
1400	4-40-4	1.89%
	4-65-4	.81%
	4-70-4	1.06%
1900	4-60-4	1.12%
	4-70-4	.92%
	4-80-4	.79%
	4-85-4	.89%

Table 3.19: All trained neural models for butterfly stub are tested with one testing data set. The tendency shows that by increasing the hidden neurons the testing error decreases until the right number of training sample beyond which no further improvement in testing error can be reached.

Figure 3.37 Testing errors of butterfly stub for (a) S_{11R} (b) S_{11I} (c) S_{12R} (d) S_{12I} . The horizontal axis is the sample number and the vertical axis is the percent testing error for the relevant S parameter.

Figure 3.38 Frequency response obtained for butterfly stub with the horizontal axis being the frequency and the vertical axis being the relevant S parameter. The width is being varied here.

3.2.6 Microstrip Width Taper Neural Models

The width taper is used in many microstrip circuit. The geometry of a taper is shown in Figure 3.39. This component connects two microstrip components with different widths together in a smoother fashion compared to discontinuities such as a step. A neural model was developed for a width taper. The model structure is shown in Figure 3.40. The inputs to the model are frequency (f), length (L), input width (w_1), output width of the taper (w_2) whose definitions and ranges are listed in Table 3.1 and Table 3.20. The outputs are S parameters - S_{11R} , S_{11I} , S_{12R} , S_{12I} , S_{21R} , S_{21I} , S_{22R} , S_{22I} . Four sets of training data and one set of testing data were created using HP Momentum analysis. The number of testing data is 2000. As usual, the training data is independent from testing data. The tendency in Table 3.21 suggests that the best trained structure for microstrip width taper is 4-90-8 trained by 2200 samples, beyond which no improvement has been reached. It's worth mentioning that all the models listed in Table 3.21 are tested with one testing data set. The testing data set is independent from training data and is never used during training phase to ensure that the low testing data results only from learning the Input/Output relation rather than "memorizing" the data points.

As Figure 3.41 suggests, the model has learned the relationship between the Input/Output. The training took about an hour to complete. The model was trained using the Quasi-Newton method following the application of a conjugate gradient training algorithm. The neural model frequency responses corresponding to different input widths are shown in Figure 3.42. EM simulations take an hour to complete, however once the model

is trained, it takes only 20 seconds for the neural model to provide the response. Therefore a good agreement has been achieved between the neural model response and that of the exact EM model in much less time compared to EM simulator.

Figure 3.39 Microstrip width taper Geometry (a) 2 dimensional (b) 3 dimensional view

Figure 3.39 Microstrip width taper Geometry (a) 2 dimensional (b) 3 dimensional view

Figure 3.40 Neural network structure for a width taper.

Name Of The Component	Frequency	w_1 (mil)	w_2 (mil)	L (mil)
Width taper	$0.5 < f < 55$	$0.8 < w_1 < 2.8$	$1 < w_2 < 2.9$	$50 < L < 150$

Table 3.20: Inputs to width taper neural model.

No of training samples	structure	test error
400	4-60-8	3.68%
	4-70-8	3.15%
	4-80-8	2.92%
	4-80-8	2.99%
1000	4-50-8	2.72%
	4-55-8	2.47%
	4-65-8	1.52%
	4-70-8	1.58%

Table 3.21: All trained neural models for width taper are tested with one testing data. The tendency shows that by increasing the hidden neurons the testing error decreases until the right number of training sample beyond which no further improvement in testing error can be reached

No of training samples	structure	test error
1700	4-60-8	1.17%
	4-68-8	.94%
	4-75-8	.79%
	4-80-8	.82%
2200	4-80-8	.92%
	4-90-8	.76%
	4-95-8	.84%

Table 3.21: All trained neural models for width taper are tested with one testing data. The tendency shows that by increasing the hidden neurons the testing error decreases until the right number of training sample beyond which no further improvement in testing error can be reached

Figure 3.41 The horizontal axis is the sample number and the vertical axis is test error.

Figure 3.41 The horizontal axis is the sample number and the vertical axis is test error.

Figure 3.42 The width taper frequency response for the appropriate S parameters (vertical axis) obtained by neural model. W_1 is being varied.

Figure 3.42 The width taper frequency response for the appropriate S parameters (vertical axis) obtained by neural model. W_1 is being varied.

Chapter 4

Applications of the Library in Optimization and Yield Analysis of Microwave Circuits

In order to judge the validity of the models obtained, two examples are presented in this chapter. The developed neural models were linked into HP-ADS [61], a software for analysis of microwave circuits. Through a neuroADS plug-in module [62], the models were connected together to construct a microstrip circuit. The resulting circuit was analyzed with S parameter analysis. Then the designs were optimized to meet certain requirements. Yield analysis was performed afterwards to examine the effect of varying some input parameters - within a certain range- on the output curves and therefore the response of the circuit. Yield is actually the percentage of the number of acceptable units to the total number of the trials based on a certain definition. In the real world the yield is important since the characteristics of the components manufactured are not exactly as is required and always has a tolerance. A small change in a component characteristic changes output

responses, depending on how sensitive the output is on that specific component parameter variation and the tolerance of the component parameters. With neural models obtained, yield analysis was performed faster than using conventional EM approach. The savings in time is even more noticeable when the number of outcomes goes higher. This is another area where it is more advantageous to use neural network approach than the time consuming EM approach.

4.1 A Microstrip Filter

In this example, a microstrip line comb filter is designed using the developed neural model library. This example is based on the circuit topology given in HPEEsof examples. The models used in this example are symmetrical T-Junction and transmission lines and short circuited stubs. Since all the components are built on the same substrate, they share common dielectric constant.

4.1.1 Simulation: Comparison with EM Simulations

First the entire circuit is simulated by EM simulator, e.g., momentum, then a schematic of equivalent circuit is constructed by neural models in design as shown in Figure 4.1. The geometrical parameters such as the lengths and widths of components can have different values for different components. In other words, the neural model is trained once but it can be called several times in a design. The length of T11, T12, Sh-st1 and Sh-st2 is 70 mil and the lengths of the rest are 150 mil. The width of T15 and T13 is 4mil and the width of other transmission lines is 3.8 mil. The input and output widths of the symmetrical T-junctions are 3.8 and 4 mil respectively. The minimum at S_{11} occurs at 15.48 GHZ

which is -45 db.

Figure 4.1: The schematic of a filter and a component neural model equivalent.

The neural model equivalent of this circuit has been constructed and linked to HP-ADS to simulate the S parameters. The results are compared to that obtained from the EM simulator, as shown in Figure 4.2 and Figure 4.3. The results from EM approach and neural model approach are pretty close which is an indication of the validity of the models. These results were provided much faster than a conventional EM simulator with an accuracy quite comparable with the EM simulation. To have a better view of the improvement the use of the models offer, a comparison is made. The simulation time for the EM simulator (HP Momentum) is about 1 hour and 8 minutes on a Sun-Sparc station, however it only took 1 minute and 45 seconds to simulate the circuit using the neural models. This indicates that once the models are trained, they can be repeatedly used in any design to provide faster results with comparable accuracy. The output S parameters using equivalent circuits in HP-ADS default library are obtained which is shown by diamonds. In reality more advanced equivalent models exist with very high accuracy.

Figure 4.2: The comparison between neural network approach and EM approach in (a) $|S_{11}|$ (dB) and in (b) $\angle S_{11}$ (deg) where the triangles represent response from physical EM approach and the solid line represents the response from neural approach and the diamonds represent the equivalent model response. The vertical axis is the appropriate S parameter and the horizontal axis is the frequency.

Figure 4.3: The comparison between neural network approach and EM approach in (a) $|S_{21}|$ (dB) and in (b) $\angle S_{21}$ (deg) where the triangles represent response from physical EM approach and the solid line represents the response from neural approach and the diamonds represent the equivalent model response. The vertical axis is the appropriate S parameter and the horizontal axis is the frequency.

4.1.2 Optimization of Design

The goal is to set the center frequency occur at 15 GHZ exactly, therefore an optimization was performed to reach this goal. In this example optimization variables include widths of the T junction and the length of the transmission lines. The optimization result show that a smaller transmission line is needed. The return loss response, S_{11} , is given in Figure 4.4 (a) and (b). After the optimization the minimum is still less than -30db. Twenty iterations were used to reach the optimal design. It took about 2 minutes to optimize the problem by neural network approach. After finding the optimal values for the components, the values were inserted in EM simulator to obtain the accurate response. The results are compared in Figure 4.4 and Figure 4.5. Also, the output S parameters using equivalent circuit models for microwave components are obtained for comparison (shown by the diamonds). The results show a close agreement between the obtained results from the EM/physical model and the neural model.

Figure 4.4: The optimized filter responses for (a) $|S_{11}|$ (dB) and b) $\angle S_{11}$ (deg). The triangles show the EM results whereas the solid lines show the neural network approach and the diamonds represent the equivalent model response. The vertical axis is the appropriate S parameter and the horizontal axis is the frequency.

Figure 4.5: The optimized filter responses for (a) $|S_{12}|$ (dB) and b) $\angle S_{12}$ (deg). The triangles show the EM results whereas the solid lines show the neural network approach and the diamonds represent the equivalent model response. The vertical axis is the appropriate S parameter and the horizontal axis is the frequency.

4.1.3 Yield Analysis

To examine the effect of changing the geometries on the center frequency, the yield analysis was performed by randomly varying geometry throughout the analysis. As is evident from Figure 4.2, the $|S_{11}|$ minimum occurs at approximately $f=15.48$ GHz. The goal is to find the number of trials whose center frequency falls within 10% of the nominal frequency, namely 13.5 ~ 16.5 GHz. The transmission line lengths are varied during 50 iterations within 15% of tolerance, and we can see that for most of the plots, the center frequency shifts about 10% of the nominal value. The yield figure was found to be 87%. It is almost impossible to use EM simulation within yield analysis since each iteration takes about 20 hours to complete so, it could take almost 2 months for 50 iterations to complete. For the case using the equivalent models, it took about 4 minutes to analyze the yield on a Sun-Sparc with yield figure being 88%, however it took about 50 seconds for the neural model to analyze. The resulting yield was compared with the analysis performed using the equivalent models which gives 83%.

Figure 4.6: The yield graphs for (a) $|S_{11}|$ (dB) and in (b) $\angle S_{11}$ (deg) for 50 iterations.

The vertical axis is the appropriate S parameter and the horizontal axis is the frequency.

Figure 4.7: The yield graphs for (a) $|S_{12}|$ (dB) and in (b) $\angle S_{12}$ (deg) for 50 iterations.

The vertical axis is the appropriate S parameter and the horizontal axis is the frequency.

To have a better view of the situation, the tolerances were unchanged and 500 trials were made. Again as is evident from the Figure 4.8 and Figure 4.9, the center frequency has shifted quite noticeably from their nominal values which makes the design quiet unaccept-

able for production since most of the designs are far from the specification if the component tolerance is 15%. The yield factor was found to be 24% with neural network models and 19% with equivalent models.

Figure 4.8: 500 iterations of (a) $|S_{11}|$ (dB) and (b) $\angle S_{12}$ (deg). The vertical axis is the appropriate S parameter and the horizontal axis is the frequency.

Figure 4.9: 500 iterations of (a) $|S_{12}|$ (dB) and (b) $\angle S_{12}$ (deg). The vertical axis is the appropriate S parameter and the horizontal axis is the frequency.

4.2 A Microstrip Signal Splitter

In this example, a signal splitter is designed using the developed library. The models used in this circuit are bends, transmission lines and T-Junctions. The lengths and widths of transmission lines T11 to T17 are 0.8 mil and 100 mil respectively and the width and length of the rest of the transmission lines are 2 mil and 150 mil respectively. The schematic of the circuit is shown below in Figure 4.10. The models which were trained before are used in this design.

(a)

(b)

Figure 4.10: The schematic for a signal splitter and a component neural model equivalent.

4.2.1 Simulation: Comparison with EM Simulations

The models were put together and linked with HP-ADS. The responses obtained from neural model approach are compared with those obtained from EM simulations as shown in Figure 4.11 and Figure 4.12. These responses show the accuracy of the model obtained from neural models. The neural model approach provides fast response to the problem.

Figure 4.11: Comparison between neural model approach versus EM approach for (a) $|S_{11}|$ (dB) and (b) $\angle S_{11}$ (deg). The triangles represent EM response and the solid line represents neural network curve and the diamonds represent the equivalent models response. The vertical axis is the appropriate S parameter and the horizontal axis is the frequency.

Figure 4.12: comparison between neural model approach versus EM approach for (a) $|S_{12}|$ (dB) and (b) $\angle S_{12}$ (deg). The triangles represent EM response and the solid line represents neural network curve and the diamonds represent the equivalent models response. The vertical axis is the appropriate S parameter and the horizontal axis is the frequency.

The output S parameters using equivalent models are shown by diamonds for comparison. These models are obtained from the default HP-ADS library which are not very accurate. However, there are more accurate equivalent models with higher accuracy.

The savings is major when the number of frequency points to be simulated is high, where for an EM simulator it takes quite a while to provide answer. This simulation takes 90 minutes for exact EM simulator however it was done instantly with the neural models (about 2 minutes) with high accuracy.

4.2.2 Optimization of the Design

The maximum return loss (S_{11}) happens at about 48.2 GHz. The objective is to set the center frequency at exactly 48 GHz. The optimization variables are transmission lines lengths and widths of T junctions. The optimization was done by the neural models obtained and the resulting responses are shown in Figure 4.13. The optimization was performed by neural models which provides a fast and accurate approach compared to EM simulation which is accurate but very slow. The optimization resulted in slightly smaller lengths. The optimization is first performed using neural models and then the obtained values for neural models(inputs) are inserted for their counterparts in an EM simulator. The results are shown in Figure 4.13 and Figure 4.14. It took only 2 minutes to complete the optimization using neural models which when compared to EM simulation time, about 90 minutes, for completing only one simulation, obviously shows the advantage of using neural models for simulation. The output S parameters using equivalent models in HP-ADS default library is also shown (by diamonds) for comparison.

Figure 4.13: The optimized responses for (a) $|S_{11}|$ (dB) and (b) $\angle S_{11}$ (deg). The triangles represent the EM response and the solid line represent the neural network response and the diamonds represent the equivalent model response. The vertical axis is the appropriate S parameter and the horizontal axis is the frequency.

Figure 4.14: The optimized responses for (a) $|S_{12}|$ (dB) and (b) $\angle S_{12}$ (deg). The triangles represent the EM response and the solid line represent the neural network response and the diamonds represent the equivalent model response. The vertical axis is the appropriate S parameter and the horizontal axis is the frequency.

4.2.3 Yield Analysis

Yield analysis was performed and the lengths of the transmission lines were changed randomly within 10% of their nominal values. The objective is to find the number of trials whose center frequencies change within 2.5% of the nominal value. The yield was found to be 83% for 50 trials. The results are shown in Figure 4.15 and Figure 4.16. It is evident from Figure 4.15 and Figure 4.16 that in this case most center frequency varies within 10% of its nominal value.

Figure 4.15: 50 trials shown for (a) $|S_{11}|$ (dB) and (b) $\angle S_{11}$ (deg). The vertical axis is the appropriate S parameter and the horizontal axis is the frequency.

Figure 4.16: 50 trials shown for (a) $|S_{12}|$ (dB) and (b) $\angle S_{12}$ (deg). The vertical axis is the appropriate S parameter and the horizontal axis is the frequency.

The yield analysis was performed again for 500 trials and the output curves shown in Figure 4.17 and Figure 4.18. The length tolerances were set to 25%. Again the neural models have provided a fast response to the problem compared to the conventional EM simulations especially if the number of the points and iterations are high. Using the neural models, it took about 4 minutes to perform yield analysis, where the yield was found to be 32%, however using the equivalent model in HP-ADS, it took about 6 minutes to perform yield analysis on a Sun-sparc machine with yield figure being 35%. In case of using accurate EM analysis, it may take months to complete the 500 iterations since just one iteration takes about 27 hours to complete. As is seen you have the accuracy of the accurate EM analysis combined with the speed of equivalent models. Therefore a more accurate result is obtained compared to the equivalent model in less time.

Figure 4.17: 500 iterations of (a) $|S_{11}|$ (dB) and $\angle S_{11}$ (deg). The vertical axis is the appropriate S parameter and the horizontal axis is the frequency.

Figure 4.18: 500 iterations of (a) $|S_{12}|$ (dB) and $\angle S_{12}$ (deg). The vertical axis is the appropriate S parameter and the horizontal axis is the frequency.

4.3 Summary

Neural models developed in previous chapters are used in circuit simulation for statistical yield analysis and optimization. A comparison between the obtained results from EM simulator and neural model is performed. This comparison shows a close agreement between the obtained results. Then an optimization is performed by using the neural model. The obtained values are then substituted for input parameters in the EM simulator. The comparison between the obtained results reemphasizes the close agreement between the neural model and exact input/output relationship. This, in turn emphasizes the close agreement between the neural model for each component and their exact input/output relationship. Then a yield analysis is performed for each design. The results are then compared with those performed with equivalent models in HP-ADS. The difference in the obtained yield figures can be attributed to the difference between the models. In fact, where the neural models represent an exact relation between input and output, the equivalent circuit models in the software fail to respond accurately.

Chapter 5

Conclusions and Remarks

In this work, a discussion of various training algorithms including the back propagation algorithm and its variants, second order training algorithms such as conjugate gradient and Quasi-Newton algorithms is provided and different neural network structures are briefly reviewed. Then a library of neural models has been developed for 10 microstrip components and two microwave circuit design examples are given. The testing errors for the neural models developed in this work are evident indications that microwave components can be precisely modeled by a neural network approach and the models can be used for fast accurate design. Then the optimization and yield analysis were carried out using the neural equivalent models for each component. The yield analysis performed for the examples shows the speed and accuracy of the neural network approach. As a result of using a neural approach, the time required to solve a problem is effectively reduced while the response is almost as accurate as the response from momentum mesh analysis, a precise, but slow method. The same approach can be used to model other microwave components and circuits, where accurate and fast models are not available yet. There are two important considerations in efficiently developing an accurate neural model to solve

microwave design problem, such as proper neural network structure and suitable training algorithms. Neural networks have a promising future in microwave design. By application of neural network methodology, all aspects of microwave design from device to components, from circuits to systems and from modeling to simulation can benefit from this robust approach. The future research work in this area will move from the demonstration of the importance of the neural network approach towards addressing the challenges in the real world microwave applications. As an example, generating training data by EM simulation is costly and the cost increases by the number of training samples, so the methodology of developing reliable neural network models with a small number of training data remains an important challenge to be addressed in future. The second important challenge that should be addressed in future is the method of dealing with errors resulting from convergence problems. These errors occur due to simulator limitations when data generation goes to extreme limits in parameter space. This situation happens in most practical cases and as such the training data are subject to large errors and, consequently, the obtained model is not reliable. Present training algorithms are sensitive to these errors which makes it extremely difficult -if not impossible - to develop a reliable model. Therefore, the development of robust training algorithms which deal with this problem is another challenge in future. One of the most important challenges in this area is how to tackle problems with highly nonlinear, non-smooth relationships between input and output parameters. The Present approach to this challenge using existing structures is to add more hidden neurons which further requires a higher number of training samples, which, as is mentioned earlier, is costly. The same problem arises when there are too many variables in the model, the number of training samples to be obtained is quite large in order to cover all different com-

binations of the variables in the parameter space. Addressing this problem is another challenge in future. The neural network approach will remain the most powerful approach, compared to all existing methods, for microwave design due to its ability to learn and generalize and its fast response. An increase of the applications of the neural network modeling approach is anticipated in the future for microwave design.

References

- [1] F. Wang and Q.J. Zhang, "Knowledge based neural models for microwave design," *IEEE Trans. microwave Theory Tech.*, vol. 45, pp.2333-2343, 1997.
- [2] P.Meijier, "Fast and smooth highly nonlinear multidimensional table models for device modeling," *IEEE Trans. Circuit Syst.*, vol. CAS-36, pp.1449-1454, 1989.
- [3] A.H. Zaabab, Q.J. Zhang, and M.S. Nakhla, "Neural network modeling approach to circuit optimization and statistical design," *IEEE Trans. Microwave Theory Tech.*, vol. 43, pp. 1349-1358, 1995.
- [4] G. Kothapalli, Artificial neural networks as aids in circuit design," *Microelectronics Journal*, vol. 26, pp. 569-678,1995.
- [5] M.Vai, S. WU, B. Li, and S. Prasad, "Creating neural based microwave circuit models for analysis and synthesis," *Proc. Asia Pacific Microwave Conf.*, pp.853-856, Hong Kong, Dec.1997.
- [6] G.L. Creech, B. Paul, C. Lesniak, T. J.Jenkins, R.Lee and M. C.Calcatera,"Artificial neural networks for microwave CAD applications,"*IEEE Intl. Microwave Symp.Digest.*, pp. 733-736, Sanfrancisco, 1996.
- [7] R.F. Harrington, *Field Computation by moment methods*, Macmillan, New York, 1968.
- [8] NeuroModeler version 1.0, Prof.Q.J.Zhang and his Neural Network Research team, Dept. of Electronics, Carleton University, 1125 Colonel By Dr., Ottawa, Ontario, K1S 5B6, Canada, 1998.
- [9] V.B. Litovski, J.I. Radjenovic, Z.M. Mrcarica, and S.L. Milenkovic, "MOS transistor

- modeling using neural network," *Electronic Letters*, vol. 28, pp. 1766-1768, 1992.
- [10] A.H. Zaabab, Q.J. Zhang, M.S. Nakhla, "Analysis and optimization of microwave circuits and devices using neural network models," *IEEE Intl. Microwave Symp. Digest*, pp. 393-396, San Diego, CA, May 1994.
- [11] G.L. Creech, B. Paul, C. Lesniak, T.J. Jenkins, R. Lee and M.C. Calcaterra, "Artificial neural networks for fast and accurate EM-CAD of microwave circuits," *IEEE Trans. Microwave Theory Tech* vol. 45, pp. 794-802, 1997.
- [12] P.M. Watson and K.C. Gupta, "EM-ANN models for microstrip vias and interconnects in data set circuits," *IEEE Trans. Microwave Theory Tech.*, vol. 44, pp. 2495-2503, 1996.
- [13] P.M. Watson, K.C. Gupta, and R.L. Mahajan, "Development of knowledge based artificial neural network models for microwave components," *IEEE Intl. Microwave Symp. Digest*, pp. 13-16, Baltimore, MD, 1998.
- [14] Y. Harkouss, J. Rousset, H. Chehade, D. Barataude, and J.P. Teyssier, "Modeling microwave devices and circuits for telecommunication system design," *Proc. IEEE Intl. Conf. Neural Networks*, pp. 128-133, Anchorage, Alaska, May 1998.
- [15] P.M. Watson and K.C. Gupta, "Design and optimization of CPW circuits using EM-ANN models for CPW components," *IEEE Trans. Microwave Theory and Tech.*, vol. 45, pp. 2515-2523, 1997.
- [16] A. Veluswami, M.S. Nakhla, and Q.J. Zhang, "The application of neural networks to EM based simulation and optimization of interconnects in high speed VLSI circuits," *IEEE Trans. Microwave Theory Tech.*, vol. 45, pp. 712-723, May 1997.
- [17] A.H. Zaabab, Q.J. Zhang, and M.S. Nakhla, "Neural Network modeling approach to

- circuit optimization and statistical design, "IEEE Trans. Microwave Theory Tech., Vol. 43 pp. 1349-1358, 1995.
- [18] Q.J. Zhang, F. Wang, 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
- [19] Q.J. Zhang and M.S. Nakhla, "Signal integrity analysis and optimization of VLSI interconnects using neural network models," *IEEE Intl. Symp. Circuits Systems*, pp. 459-462, London, England, 1994.
- [20] P. Burrascano, M. Dionigi, C. Fancelli, and M. Mongiardo, "A neural network model for CAD and optimization of microwave filters," *IEEE Intl. Microwave Symp. Digest*, pp. 13-16, Baltimore, MD, 1998.
- [21] M.D. Baker, C.D. Himmel, and G.S. May, "In-Situ prediction of reactive ion etch end point using neural networks," *IEEE Trans. Components, Packaging, and Manufacturing Tech., part A*, vol. 18, pp. 478-483, Sept. 1995.
- [22] M. Vai and S. Prasad, "Microwave circuit analysis and design by a massively distributed computing network," *IEEE Trans. Microwave Theory Tech.*, vol. 43, pp. 1087-1094, May 1995.
- [23] M. Vai and S. Prasad, "Automatic impedance matching with a neural network," *IEEE Microwave Guided Wave Letter*, vol. 3, pp. 353-354, 1993.
- [24] R. Biernacki, J.W. Bandler, J. Song, and Q.J. Zhang, "Efficient quadratic approximation for statistical design," *IEEE Trans. Circuits Syst.*, vol. CAS-36, pp. 1449-1454, 1989.
- [25] J. Park and I.W. Sandberg, "Universal approximation using Radial-Basis-Function networks," *Neural Computation*, vol. 3, pp. 246-257, 1991.

- [26]J. Park and I.W. Sandberg, "Approximation and Radial-Basis-Function networks," *Neural Computation*, vol.5, pp. 305-316, 1993.
- [27]A. Krzyzak, T. Linder, and G. Lugosi, "Nonparametric estimation and classification using radial basis function nets and empirical risk minimization, "*IEEE Trans. Neural Networks*, vol. 7, pp. 475-487, Mar. 1996.
- [28] G.G. Towell and J.W. Shavlik, "knowledge-based artificial neural networks," *Artif. Intell.*, vol. 70, pp. 119-165, 1994.
- [29] R. Maclin and J.W. Shavlik, "Using knowledge based neural networks to improve algorithms: refining the Chou-Fasman algorithm for protein folding," *Machine Learning*, vol. 11, pp. 195-215, 1993.
- [30].M. Fu, "Integration of neural heuristics into knowledge based inference," *Connection Science*, vol. 1, pp. 325-340, 1989.
- [31]R.C. Lacher, S.I. Hruska, and D.C. Kuncicky, "Back-Propagation learning in expert networks, "*IEEE Trans. Neural Networks*, vol. 3, pp. 62-72, Jan. 1992.
- [32]J.J. Mahoney and R.J. Mooney, "Combining connectionist and symbolic learning to refine certainly factor rule bases," *Connection Science*, vol. 5, pp. 339-364, 1993.
- [33]W.J. Jansen, M.Diepenhorst, J.A.G. Nijhuis, and L.Spaanenburg, Assembling engineering knowledge in a modular multi-layer perceptron neural network, "*Proc. IEEE Intl. Conf. Neural Networks*, pp. 232-237, Houston, Tx, June 1997.
- [34] G.G. Lendaris, A. Rest and T.R. Misley, Improving ANN generalization using a priori knowledge to pre-structure ANNs, "*Proc. IEEE Intl. Conf. Neural Networks*, pp. 248-253, Houston, TX, June 1997.
- [35]J.R.Tsai, P.C. Chung, and C.I.Chang, "A sigmoidal radial basis function neural net-

- work for function approximation," *Proc. IEEE Intl. Conf. Neural Networks*, pp. 469-501, Washington, DC, June 1996.
- [36] Q.H. Zhang, "Using wavelet network in nonparametric estimation," *IEEE Trans. Neural Networks*, vol. 8, pp. 227-236, Mar. 1997.
- [37] Q.H. Zhang and A. Benveniste, "Wavelet networks," *IEEE Trans. Neural Networks*, vol. 3, pp. 889-898, Nov. 1992.
- [38] B.R. Bakhshi and G. Stephanopoulos, "Wave-Net: a multiresolution, hierarchical neural network with localized learning," *American Institute of Chemical Engineering Journal*, vol. 39, pp. 57-81, 1993.
- [39] B. Delyon, A. Juditsky, and A. Benveniste, "Accuracy analysis for wavelet approximations," *IEEE Trans. Neural Networks*, vol. 6, pp. 332-348, Nov. 1995.
- [40] S.E. Fahlman and C. Lebiere, "The cascade-correlation learning architecture," *Advances in neural information processing*, vol. 2, D.S. Truettzkey, Eds. San Mateo, CA: Morgan Kaufmann, 1990, pp. 524-532.
- [41] N.K. Treadgold and T.D. Gedeon, "Extending CasPer: a regression survey," *Intl. Conf. Neural Information Processing*, pp. 524-532.
- [42] N.K. Treadgold and T.D. Gedeon, "Exploring architecture variation in constructive cascade networks," *Proc. IEEE Intl. Conf. Neural Networks*, pp. 343-348, Anchorage, Alaska, May 1998.
- [43] J.N. Hwang, S.S. You, S.R. Lay, and I.C. Jou, "The Cascade-Correlation learning: a projection pursuit learning perspective," *IEEE Trans. Neural Networks*, vol. 7, pp. 278-289, 1996.
- [44] D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning internal representations by

- error propagation," In parallel Distributed Processing, vol. I, D.E. Rumelhart and J.L. McClelland Eds. Cambridge, Massachusetts: MIT Press, 1986, pp.318-362.
- [45]W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling, Numerical Recipes: The Art Of Scientific Computing, Cambridge, Massachusetts: Cambridge University Press, 1986.
- [46]W. Kaminski and P. Strumilo, "Kernel orthonormalization in radial basis function neural networks, "IEEE Trans. Neural Networks, vol. 8, pp. 1177-1183, 1997.
- [47]S. Haykin, Neural networks: A comprehensive foundation, Newyork, NY: IEEE press, 1994.
- [48]M.F. Moller, "A scaled conjugate gradient algorithm for fast supervised learning,"*Neural Networks*, vol. 6, pp. 525-533, 1993.
- [49] T.R. Cuthbert Jr., "Quasi-Newton methods and constraints,"In *Optimization using Personal Computers*, Newyork, NY: John Wiley & Sons, 1987, pp.233-314.
- [50]A.J. Shepherd, "Second-order optimization methods," In Second-Order Methods for Neural Networks, Berlin, NY: Springer-Verlag, 1997, pp. 43-72.
- [51]S. Kollias and D. Anastassiou, "An adaptive least squares algorithm for the efficient training of artificial neural networks,"*IEEE Trans. Circuits and Systems*, 1996, vol. 36, pp. 43-72.
- [52]M.G.Bello, "Enhanced training algorithms, and integrated training/architecture selection for multilayer perceptron networks," *IEEE Trans. Neural Networks*, vol. 3, pp. 864-875, Nov. 1992.
- [53]S. Ergezinger and E. Thomsen, "An accelerated learning algorithm for multilayer perceptrons: Optimization layer by layer, *IEEE Trans. Neural Networks*, vol. 6 PP. 31-

42, Jan. 1995.

- [4] G.J. Wang and C.C. Chen, "A fast multilayer neural network training algorithm based on layer by layer optimizing procedures," *IEEE Trans. Neural Networks*, vol. 7, pp. 768-775, 1996.
- [55] R. Lengelle and T. Denceux, "Training MLPs layer by layer using an objective function for internal representations," *Neural Networks*, vol. 9, pp. 83-97, 1996.
- [56] J.S. Shawe-Taylor and D.A. Cohen, "Linear programming algorithm for neural networks," *Neural Networks*, vol. 3, pp. 575-582, 1990.
- [57] S. Mcloone, M.D. Brown, G. Irwin, and Lightbody, A hybrid linear/nonlinear training algorithm for feedforward neural networks, "IEEE Trans. Neural Networks, vol. 9, pp. 669-684, July 1998.
- [58] S. Kirkpatrick, C.D. Gellatt, and M.P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671-680, 1983.
- [59] J.C.F. Pujoul and R. Poli, Evolving neural networks using a dual representation with a combined crossover operator," *Proc. IEEE Intl. Conf. Evol. Comp.*, pp. 416-421, Anchorage, Alaska, May 1998.
- [60] T. Rognvaldson, "On Langevin updating in multilayer perceptron," *Neural Computation*, vol. 6, pp. 916-926, 1994.
- [61] Advanced Design System HPEESOFSIM ver. 110 rev. 300, by Hewlett-Packard company, 3000 Hanover Street, Palo Alto, CA 94304, USA, 1999.
- [62] NeuroADS version 1.2, Prof.Q.J.Zhang and his Neural Network Research team, Dept. of Electronics, Carleton University, 1125 Colonel By Dr., Ottawa, Ontario, K1S 5B6, Canada, 1999.