Parametric Modeling of EM Behaviors of Microwave Components Using Combined Neural Networks and Pole-Residue Transfer Functions

by

Feng Feng, B.Eng

A thesis submitted to

the Faculty of Graduate and Postdoctoral Affairs

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Electronic Science and Technology, School of Microelectronics

Tianjin University, Tianjin, China

Ottawa-Carleton Institute for Electrical and Computer Engineering

Carleton University, Ottawa, Ontario, Canada

Abstract

Parametric modeling of electromagnetic (EM) behaviors has become important for EM design optimizations of microwave components. The EM based design, such as design optimization, what if analysis and yield-driven design, can be time consuming because it usually requires repetitive EM simulations with varying values of geometrical parameters as design variables. Parametric models can be developed from the information of EM responses as functions of geometrical parameters. The developed parametric models allow faster simulations and optimizations with varying values of geometrical parameters and subsequently can be implemented in high-level circuit and system design optimizations.

This thesis proposes a novel technique to develop combined neural network and pole-residue-based transfer function models for parametric modeling of EM behaviors of microwave components. In this technique, neural networks are trained to learn the relationships between pole/residues of the transfer functions and geometrical parameters. The orders of the pole-residue transfer functions may vary over different regions of geometrical parameters. We develop a pole-residue tracking technique to solve this order-changing problem. After the proposed modeling process, the trained model can be used to provide accurate and fast predictions of the EM behavior of microwave components with geometrical parameters as variables.

An advanced pole-residue tracking technique is proposed to exploit sensitivity information to solve the challenges of pole-residue tracking especially when the amount of training data are reduced and/or the geometrical step sizes between the data samples are enlarged. The proposed technique takes advantages of sensitivity

information to split one pole into two separate new poles to achieve the increase of the orders of the transfer functions and ultimately form transfer functions of a constant order over the entire region of geometrical parameters. The proposed technique addresses the challenges of pole-residue tracking when training data are limited.

As a further advancement, we introduce EM sensitivity analysis into the pole-residue-based neuro-transfer function modeling technique. The purpose is to increase the model accuracy by utilizing EM sensitivity information and to speedup the model development process by reducing the number of training data required for developing the model. The proposed parametric model consists of the original and adjoint pole-residue based neuro-TF models. New formulations are derived for calculating the second order derivatives for training the adjoint pole-residue based neuro-TF model. By exploiting the sensitivity information, the proposed technique can further speed up the model development process over the existing pole-residue parametric modeling method without using sensitivity analysis. The proposed parametric modeling techniques in this thesis are demonstrated by several microwave examples.

KEY WORDS: Electromagnetic, parametric modeling, neural network, transfer function, sensitivity analysis.

To my parents

Shuli Peng and Fushan Feng

Acknowledgements

I would like to express my sincere appreciation to my supervisor, Prof. Q. J. Zhang for the constant support and expert guidance during the course of this research work. His encouragement, motivation and thoughtful insights during my research made my journey in realizing my goal a memorable one. It was my honor to work under his supervision and guidance.

I would also like to express my appreciation to my co-supervisor, Prof. Jianguo Ma for the invaluable counsel and knowledgeable instruction. His enthusiasm and professional advices provided a great help for me on my way pursuing my Ph.D degree. I would like to thank Dr. Tom Dhaene, Dr. Adrian Chan, Dr. Tet Yeap, Dr. Guoxuan Qin, Dr. Pavan Gunupudi, Dr. Haipeng Fu, Dr. Michel Nakhla, and Dr. Mustapha Yagoub for their valuable suggestions and corrections for my thesis.

I would like to thank Tianjin University, where I have stayed for nine years accomplishing both my undergraduate and Ph.D studies. I would also like to thank Carleton University for the Cotutelle program for my Ph.D study.

I would like to thank all my present and former colleagues especially Dr. Lin Zhu, Dr. Shuxia Yan, Dr. Seyed Alireza Sadrossadat, Dr. Venu-Madhav-Reddy Gongal-Reddy, Chuan Zhang, Hui Lin, Shunlu Zhang, Weicong Na, Wenyuan Liu, Chao Zhang, Kai Ma, Bo Peng, Jianan Zhang, Wei Zhang, Zhihao Zhao and Jing Jin in our research group for their enthusiasm and valuable discussions.

I would like to thank my Ph.D roommates and friends Dr. Shihui Yu, Dr. Bo Wang, Dr. Heping Shi and Dr. Qianfu Cheng for their assistance and advices. I would also like to thank my friends Tianyu Xiang, Panbo Yang, Qiushi Li, Yao

Song, Ning He, Yitong Zhang and Jing Yu for the helps and memorable experiences during the life in Tianjin.

Finally, I wish to thank my parents, my grandparents, my uncles and aunts, my brothers and sisters for their love, belief, support and encouragement throughout my study. A special thank to my dear friend Li Ma for her constant accompany, encouragement and motivation.

Table of Contents

List of Figures

List of Tables

List of Symbols

$\boldsymbol{A}_k$	The data matrix for the derivatives of poles and residues with respect to geometrical parameters $\boldsymbol{x}$ for the k^{th} training sample
$\hat{\boldsymbol{A}}_k$	The reformatted data matrix for the derivatives of poles and residues with respect to geometrical parameters $\boldsymbol{x}$ for the k^{th} training sample after the proposed pole-residue tracking technique
a_j	The coefficients of the numerator of the transfer function
$\boldsymbol{B}_{SM}$	The scaling coefficients of a linear space mapping function
b_j	The coefficients of the denominator of the transfer function
$\boldsymbol{C}_k$	The vector containing the poles and residues of the transfer function for the k^{th} geometrical sample after the pole-residue tracking process
$\boldsymbol{c}$	The vector containing all the neural network outputs, i.e., $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\eta}$, and $\boldsymbol{\xi}$

$\tilde{\boldsymbol{c}}$	The data vector containing the real and imaginary parts of poles and residues whose order may vary among different geometrical samples
$\boldsymbol{c}_k$	The vector containing the poles and residues of the transfer function for the k^{th} geometrical sample
$\hat{\boldsymbol{c}}_k$	The reformatted data vector containing the real and imaginary parts of poles and residues for the k^{th} training sample after the proposed pole-residue tracking technique
$\boldsymbol{c}_{SM}$	The shifting coefficients of a linear space mapping function
D_m	The sum of differences between the imaginary parts of the present poles after splitting and the poles in the neighboring sample, after the trial of splitting the m^{th} pole
$\boldsymbol{d}$	The vector representing the outputs of the EM simulations (e.g., real and imaginary parts of S-parameters)
$\boldsymbol{d}'$	The derivative data of the outputs with respect to geometrical parameters from EM sensitivity analysis
$d\boldsymbol{y}/d\boldsymbol{x}$	The outputs of the adjoint neuro-TF model, i.e., the derivatives of the original neuro-TF model outputs with respect to the geometrical inputs
E_{adj}	The training error of the adjoint neuro-TF models

E_{orig}	The training error of the original neuro-TF models
E_{Pre}	The error function for the preliminary training process
E_t	The user defined threshold error
E_{Te}	The error between the model response and the testing data
E_{Tr}	The error between the model response and the training data
$\boldsymbol{F}_k$	The derivative data of the EM outputs with respect to the real and imaginary parts of poles and residues for the k^{th} training sample, containing both the real and imaginary parts of $\boldsymbol{f}_q^k$ at all the frequency points
$\boldsymbol{f}_{ANN}$	The neural network function
$\boldsymbol{f}_q^k$	The the derivatives of the transfer function response H_q^k with respect to the real and imaginary parts of all the poles and residues at the q^{th} frequency for the k^{th} sample
$\boldsymbol{G}_k$	The derivative data of the EM outputs with respect to $\boldsymbol{x}$ for the k^{th} training sample, containing both the real and imaginary parts of $\boldsymbol{g}_q^k$ at all the frequency points
$\boldsymbol{g}_q^k$	The derivative data of the EM response H_q^k with respect to geometrical parameters $\boldsymbol{x}$ at the q^{th} frequency for the k^{th} sample
$\boldsymbol{H}$	The frequency response of transfer function

H_q^k	The transfer function response at the q^{th} frequency for the k^{th} sample
K	The index of this selected sample which needs to be split
k	The index of training samples of geometrical parameters
L	The index of the neighboring geometrical sample which has the shortest geometrical distance to the k^{th} geometrical sample
N	The order of the transfer function
N_k	The effective order of the transfer function for the k^{th} geometrical sample
$N_{\max}$	The maximum effective order of the transfer function
$N_{\min}$	The minimum effective order of the transfer function
n_f	The total number of frequencies
n_k	The order of transfer function for the k^{th} geometrical sample
n_m	The number of samples in subset T_s
n_s	The total number of training samples of geometrical parameters
n_x	The total number of inputs in vector $\boldsymbol{x}$
n_y	The total number of outputs in vector $\boldsymbol{y}$

M	The index of the pole which is selected to be split
$\widehat{M}$	The index of the pole which is selected to be split using the advanced pole-residue tracking technique with sensitivity analysis
$\boldsymbol{P}^{(k)}$	The vector of poles of the transfer function for the k^{th} geometrical sample after the pole-residue tracking process
$\boldsymbol{P}_{corr}$	The added correction term at the output space in surrogate model
$\boldsymbol{P}_{SM}$	The space mapping function
$\boldsymbol{p}$	The poles of the transfer function
$\boldsymbol{p}^{(k)}$	The vector of poles for the k^{th} geometrical sample
p'_j	The poles after splitting and sorting during each pole-residue tracking iteration
p_{NN}	The outputs of the neural network for poles corresponding to geometrical variables and neural network weights
$\boldsymbol{Q}_{SM}$	The implicit space mapping function
q	The index of frequencies
$\boldsymbol{R}^{(k)}$	The vector of residues of the transfer function for the k^{th} geometrical sample after the pole-residue tracking process

$\boldsymbol{R}_c(\boldsymbol{x}_c)$	The response vector of a coarse model corresponding to a vector of coarse model design variables $\boldsymbol{x}_c$
$\boldsymbol{R}_f(\boldsymbol{x}_f)$	The response vector of a fine model corresponding to a vector of design variables $\boldsymbol{x}_f$
$\boldsymbol{R}_s(\boldsymbol{x}_f)$	The response vector of surrogate model corresponding to a vector of design variables $\boldsymbol{x}_f$
$\boldsymbol{r}$	The residues of the transfer function
$\boldsymbol{r}^{(k)}$	The vector of residues for the k^{th} geometrical sample
r'_j	The residues after splitting and sorting during each pole-residue tracking iteration
r_{NN}	The outputs of the neural network for residues corresponding to geometrical variables and neural network weights
s_q	The symbol representing frequency in the transfer function
T_r	The index set of training samples of geometrical parameters
T_s	The sub-set containing all the training samples whose orders are equal to $N_{\min}$
$\boldsymbol{w}$	The weighting parameters in the neural networks
$\boldsymbol{w}_p$	The weights of the neural network for poles

$\boldsymbol{w}_r$	The weights of the neural network for residues
$\boldsymbol{x}$	The vector of inputs of the model, containing the geometrical variables
$\boldsymbol{x}_{aux}$	The auxiliary parameters (i.e., preassigned parameters)
$\boldsymbol{x}_c$	The vector of design variables in coarse model
$\boldsymbol{x}_f$	The vector of design variables in fine model
$\boldsymbol{x}_f^*$	The optimal fine model solution
$\boldsymbol{y}$	The vector of the outputs of the model, containing real and imaginary parts of the transfer function response
Z_m	The difference of the derivatives for all the geometrical parameters between the m^{th} and $(m+1)^{th}$ poles in the L^{th} sample
$\boldsymbol{\alpha}$	The vector containing the variables of the real parts of the poles
$\tilde{\boldsymbol{\alpha}}$	The vector containing the data of the real parts of the poles whose order may vary among different geometrical samples
$\boldsymbol{\beta}$	The vector containing the variables of the imaginary parts of the poles
$\tilde{\boldsymbol{\beta}}$	The vector containing the data of the imaginary parts of the poles whose order may vary among different geometrical samples

$\boldsymbol{\eta}$	The vector containing the variables of the real parts of the residues
$\tilde{\boldsymbol{\eta}}$	The vector containing the data of the real parts of the residues whose order may vary among different geometrical samples
$\boldsymbol{\xi}$	The vector containing the variables of the imaginary parts of the residues
$\tilde{\boldsymbol{\xi}}$	The vector containing the data of the imaginary parts of the residues whose order may vary among different geometrical samples
δ_i	The smallest variance of x_i among all the samples
ϵ	A user defined threshold which decides whether D_m is close to D_M
$\boldsymbol{\Gamma}$	A vector containing the indices of the geometrical samples which have the lowest order of transfer function among all the geometrical samples
Ω	The index set of frequency samples
$\boldsymbol{\Phi}$	The index set containing the indices of poles whose sum of differences D_m is close to D_M
ϕ	The symbol used to simplify the representation of the elements in the matrix of second-order derivatives of transfer function response with respect to the real and imaginary parts of poles and residues
$*$	The superscript representing complex conjugate

$\|\cdot\|$	L_2 norm
$\|\cdot\|_F$	Frobenius norm

Chapter 1

Introduction

1.1 Introduction and Motivation

Nowadays, as the frequency of microwave circuits increases from gigahertz to terahertz, the wavelengths become even smaller. When the sizes of the devices are comparable to the operating wavelength, the three-dimensional EM simulations become important for describing the characteristics of the microwave devices. There are three important reasons to simulate EM behaviors of microwave circuits and systems: to understand the physics of a complex system of interacting elements; to verify new concepts; and to optimize designs.

EM based design, such as design optimization, what if analysis and yield-driven design, can be time consuming because it usually requires repetitive EM simulations with geometrical parameters as design variables. To speedup the process of obtaining EM responses, parametric modeling of electromagnetic (EM) behaviors has become important for EM design optimizations of microwave components. Parametric models can be developed from the information of EM responses as functions of ge-

ometrical parameters. The developed parametric models allow faster simulations and optimizations with varying values of geometrical parameters and subsequently can be implemented in high-level circuit and system designs [1]−[5].

Artificial neural network (ANN) has been a recognized vehicle for the EM parametric modeling and design optimization [1]−[12]. ANN can represent general nonlinear relationships between EM behaviors of the microwave components and the geometrical parameters after a proper training process. As a further development of ANN parametric modeling, the sensitivity analysis has been used to speed up the model development process by utilizing EM sensitivity information to reduce the number of training data required for developing the model [13]−[15]. To improve the accuracy and reliability of the EM parametric modeling and optimization, the knowledge-based/space-mapping modeling method is developed [16]−[18]. Different space-mapping techniques have been introduced to map the prior knowledge such as coarse models onto EM behaviors of microwave components [19]−[25].

An advanced modeling approach, which combines neural networks and transfer functions (neuro-TF), was developed to perform parametric modeling of EM responses [26], [27]. This approach can be used even if accurate equivalent circuits or empirical models are unavailable. In this method, transfer functions are used to represent the EM responses of passive components versus frequency. However, the coefficients of transfer functions have discontinuity problems when there are large geometrical variations. In the reference [28], a training method for developing parametric neuro-TF model of microwave passive components in the frequency domain has been presented under the rational transfer function formulation. The solution

of the discontinuity of coefficients in transfer functions over the geometrical variables is addressed in the reference [28]. The sensitivities of the coefficients of the rational transfer function with respect to geometrical parameters become high when the order of transfer function is high and/or the geometrical variations are large. This limits the accuracy and robustness of the model. Reference [29] presented an alternative pole-residue approach to the transfer function. However, the method to address discontinuity in the reference [28] applicable to rational functions is not applicable to pole-residue-based transfer functions, which requires a quite different approach. The order-changing problem for the pole/residue formulation remains an open problem when the geometrical variations are large.

In this thesis, a novel parametric modeling technique combining neural networks and pole-residue-based transfer functions is proposed to develop parametric models of EM behaviors for microwave components. This new method is proposed to address the discontinuity issue of the pole-residue-based transfer function. A novel order-changing technique is developed for pole-residue-based transfer functions to handle the problem of order variations as geometrical parameters change in a large region. As a further advancement, we also introduce sensitivity analysis into the proposed pole-residue-based neuro-transfer function parametric modeling technique. The model developed by the proposed method can obtain good accuracy with fewer training data. After the proposed modeling process, the trained model can be used to provide accurate and fast prediction of the EM responses with respect to the geometrical variables and can be subsequently used in the high-level circuit and system design.

1.2 List of Contributions

The main objective of this thesis is to investigate and develop an accurate and robust parametric modeling method of EM behavior of microwave components. In this thesis, a novel parametric model is developed addressing the condition that no coarse model are available. In addition, the developed model should be formulated such that it can be conveniently incorporated into existing circuit simulators for high-level circuit simulations and design optimizations. In this thesis, the following significant contributions are made:

- A novel technique is proposed to develop a combined neural network and pole-residue-based transfer function (pole-residue-based neuro-TF) model for parametric modeling of EM behaviors of microwave components [30]. A pole-residue format of transfer function is used to represent the frequency response of the EM behaviors of microwave components addressing the condition with no available coarse model. Neural networks are used to establish the nonlinear mapping relationship between geometrical variables and pole/residues of transfer function. An order-changing technique is developed for pole-residue-based transfer functions to address the discontinuity issue of the pole-residue-based transfer function. The accuracy of the model developed by the proposed method remains good even when the order of the model is high. After the proposed modeling process, the developed model can be used to provide accurate and fast predictions of the EM responses with respect to the geometrical variables and can be subsequently used in the high-level circuit and system

design.

- An advanced pole-residue tracking technique is proposed to exploit sensitivity information to solve the challenges of the pole-residue tracking especially when the amount of training data are reduced and/or the geometrical step sizes between the data samples are enlarged [31]. The proposed technique takes advantages of sensitivity information to split one pole into two separate new poles to achieve the increase of the orders of the transfer functions and ultimately form transfer functions of a constant order over the entire region of geometrical parameters. The proposed technique addresses the challenges of pole-residue tracking when training data are limited.

- For the first time, sensitivity analysis is introduced into the neuro-transfer function parametric modeling method [31], [32]. In this proposed technique, not only the input-output behavior of the modeling problem but also the sensitivity analysis information generated from EM simulators are used in the model development. New formulations are derived to calculate the second-order derivatives required for developing the proposed sensitivity-analysis based neuro-TF model. By exploiting sensitivity information for training, the proposed modeling method can obtain accurate parametric models with fewer training data (or obtain better accuracy with the same amount of training data).

1.3 Thesis Organization

The rest of the thesis is organized as follows:

Chapter 2 presents a literature review of popularly used parametric modeling technique for EM behavior modeling. An overview of artificial neural network (ANN), ANN with sensitivity analysis, and knowledge based neural network techniques are presented. Classical space mapping methods such as implicit space mapping, output space mapping, tuning space mapping and the recently developed coarse and fine mesh space mapping, parallel space mapping, portable space mapping are also reviewed. Recent neuro-transfer function parametric modeling method are also discussed.

Chapter 3 presents a novel technique is proposed to develop a combined neural network and pole-residue-based transfer function (pole-residue-based neuro-TF) model for parametric modeling of EM behavior of microwave components. This proposed method addresses the condition that no coarse model are available. Three application examples are used to illustrate the proposed technique.

Chapter 4 presents an advanced pole-residue tracking technique using EM sensitivity analysis. The purpose is to exploit the sensitivity information to track the splitting of poles as geometrical parameters change. The proposed technique addresses the challenges of tracking pole splitting when training data are limited. This technique is illustrated using a coupled-line filter example.

Chapter 5 presents a further advancement, which introduce sensitivity analysis into the proposed pole-residue-based neuro-transfer function parametric modeling

method. By exploiting sensitivity information for training, the proposed modeling method can obtain accurate parametric models of wider geometrical range with the same amount of training data. This proposed technique is demonstrated by three microwave examples.

Finally, Chapter 6 presents the conclusions and discussions on possible directions for future work.

Chapter 2

Literature Review

Parametric modeling of electromagnetic (EM) behavior has become important for EM design optimization of microwave components. The EM based design can be time consuming because it usually requires repetitive EM simulations with geometrical parameters as design variables. Parametric models can be developed from the information of EM responses as functions of geometrical parameters. The developed parametric models allow faster simulations and optimizations with varying values of geometrical parameters and subsequently can be implemented in the high-level circuit and system design optimizations. A variety of parametric modeling techniques have been introduced for modeling EM behavior of passive and active components. Here reviews frequently used parametric modeling techniques.

2.1 Artificial Neural Network

Artificial neural networks (ANNs) are recognized vehicles for electromagnetic (EM)-based modeling and optimization in microwave area [1]−[12]. EM design optimization can be expensive because it requires repetitive EM simulations due to adjust-

ments of the values of geometrical parameters. Through an automated training process, ANN can learn the relationship between EM responses and geometrical parameters. The trained model provides accurate and fast prediction of the EM behavior of microwave components with geometrical parameters as variables, and can be subsequently implemented in high-level circuit and system design [1].

In the past decades, significant advances have been made in development of artificial neural network to overcome RF/microwave modeling and design difficulties [2]. Neural network can learn and represent the general input-output behavior of a device/component even when the detailed internal formulas of the device/component are unavailable. Once developed, these neural network models can be used instead of computationally expensive EM models of passive/active components to speed up RF/microwave circuit design [4], [33]−[36].

Researchers have investigated a variety of important applications utilizing the ability of artificial neural networks to perform parametric modeling and optimization of microwave components and circuits, such as high-speed VLSI interconnects [37], [38], bends [17], [39], vias [40], spiral inductors [36], [41], microwave FETs [35], [42], HBTs [43], [44], HEMTs [45], waveguides [46], laser diodes [47], filters [48]−[50], amplifiers [4], [51], mixers [51], antennas [52], coplanar waveguide (CPW) components [27], [53], embedded passives [26], [54], multilayer circuit packages [55].

A variety of ANN structures have been developed in recent days [56]−[67], such as multilayer perceptron neural networks [62], recurrent neural networks [51], [58], [60], [61], time-delay neural networks [8], radial basis function neural networks [9], [63], dynamic neural networks [56], [57], and the recently introduced state-space

dynamic neural networks (SSDNN) [64]−[67]. In the references [16], [68]−[70], automated model generation method is described to automate the structure selection process of the ANN modeling.

As a further approach, the knowledge-based neural network has been developed to improve the accuracy and reliability of modeling and design optimization [35]. In the knowledge-based approach, models are developed using neural networks combined with prior knowledge such as analytical expressions [16], empirical models [17] or equivalent circuits [35], [71]. The idea of the knowledge-based model is to exploit existing knowledge in the form of empirical or equivalent circuit models together with neural networks to develop a faster and more accurate model. For the microwave design, there exist many empirical or equivalent circuit models that are computationally efficient and widely used in practical design. However, such models are often valid only in a limited parameter range, beyond which the model predictions become inaccurate.

2.1.1 Neural Network Structures

A neural network has at least two physical components, which are the processing elements and the connections between them. The processing elements are named neurons while the connections between the processing elements are called links. Every link has a weighting parameter. Each neuron receives the stimulus from the other neurons which are connected to it, processes the information and generates an output response. Neurons who receive stimulus from the outside of the neural networks are named input neurons. Neurons whose outputs are used externally

are named output neurons. Neurons who receive stimulus from other neurons and whose output is a stimulus for other neurons in the neural network are named hidden neurons. Different neural network structures can be constructed by using different processing elements or by the different manner how they are connected.

Multilayer Perceptrons (MLP) are the most popular type of neural networks, which are used in many different applications. In the structure of MLP, the neurons are divided into different layers. The first layer is called input layer and the last layer is called output layer. The rest of layers in between of these two layers are called hidden layers. In general, an MLP consists of one input layer, one or more hidden layers, and one output layer, as shown in Fig. 2.1.

According to the universal approximation theorem for MLP proved by Cybenko [72] and Hornik et al. [73], a three layer perceptron (a perceptron is defined as an algorithm for supervised learning) provided by enough hidden neurons, is able to approximate any arbitrary nonlinear function with any desired accuracy. In practice, how to choose the exact number of hidden neurons needed for specific modeling problem remains an open question. The ongoing research in this direction includes algorithms such as constructive algorithms [74], network pruning [75], regularization [76], and automatic model generation [69].

Practically, three-layer and four-layer perceptrons are most commonly used in neural network applications. In general, four-layer perceptrons have better performances in modeling higher nonlinear problem than three-layer perceptrons which may need too many hidden neurons. Three-layer perceptrons are usually preferred when generalization capability of the function approximation is a major concern

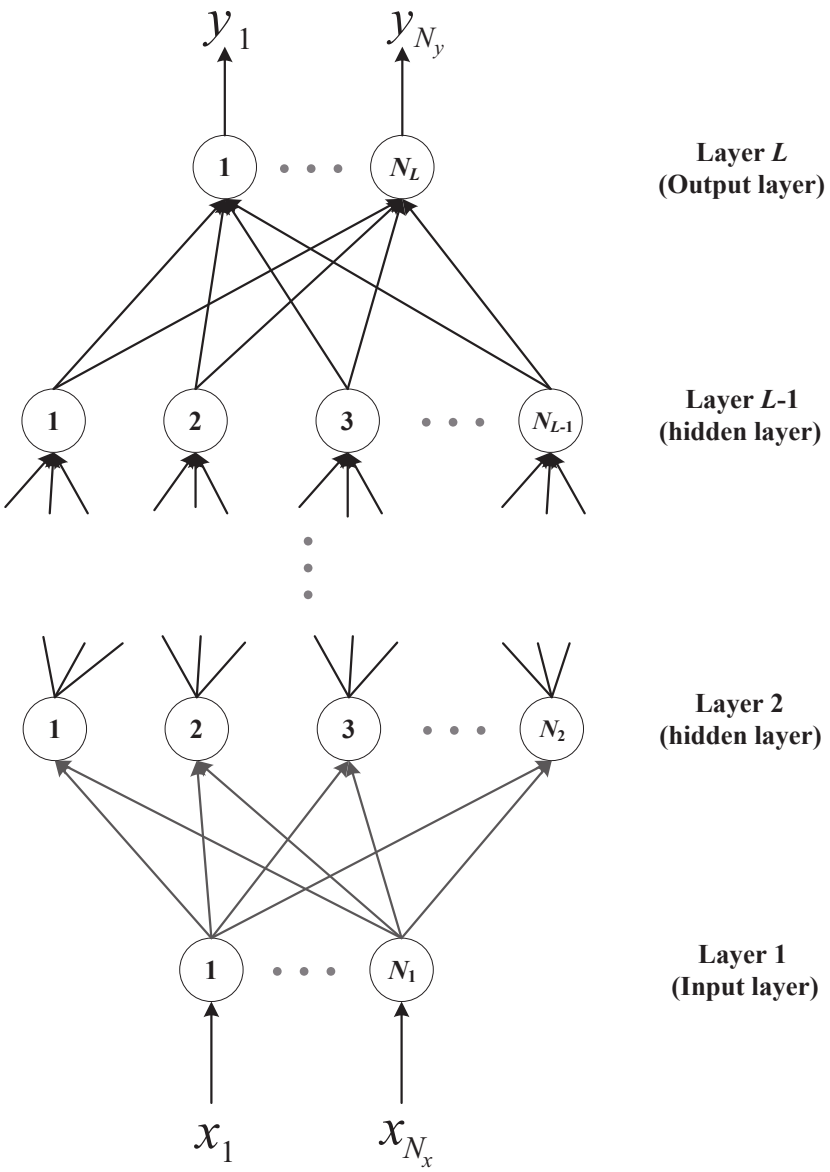


Figure 2.1: Multilayer perceptrons (MLP) structure containing one input layer, one output layer, and several hidden layers.

[77], because fewer hidden neurons are needed in the resulting neural network. Four-layer perceptrons perform better in boundary definitions which usually preferred in pattern classification problems [78].

Radial basis function (RBF) neural network is another popular neural network structure which has a single hidden layer using radial basis activation functions for hidden neurons [63], [79]. A typical RBF neural network is illustrated in Fig. 2.2. It consists of one input layer, one output layer, and one RBF hidden layer. The Gaussian and multiquadratic functions are the most common radial basis activation functions used in RBF neural networks.

The idea of combining wavelet theory with neural networks results in another type of neural network structure, called wavelet networks [80]−[82]. Wavelet functions are used as hidden neuron activation function in the wavelet networks. Network construction methods are developed using theoretical features of the wavelet transform. These methods help to determine the number of hidden neurons and the weight parameters during training process. The wavelet network has been used in modeling passive and active microwave components.

One specific type of neural network which allows time-domain behaviors of a dynamic system is described. This type of neural network is established depending not only on the present inputs but also on the history of the system states and inputs, as called recurrent neural network (RNN) [83]−[85]. A recurrent neural network structure with feedback of delayed neural network is shown in Fig. 2.3.

The combined history of the inputs and outputs of the system forms a vector of extra inputs to be presented to the neural network which can be any of the

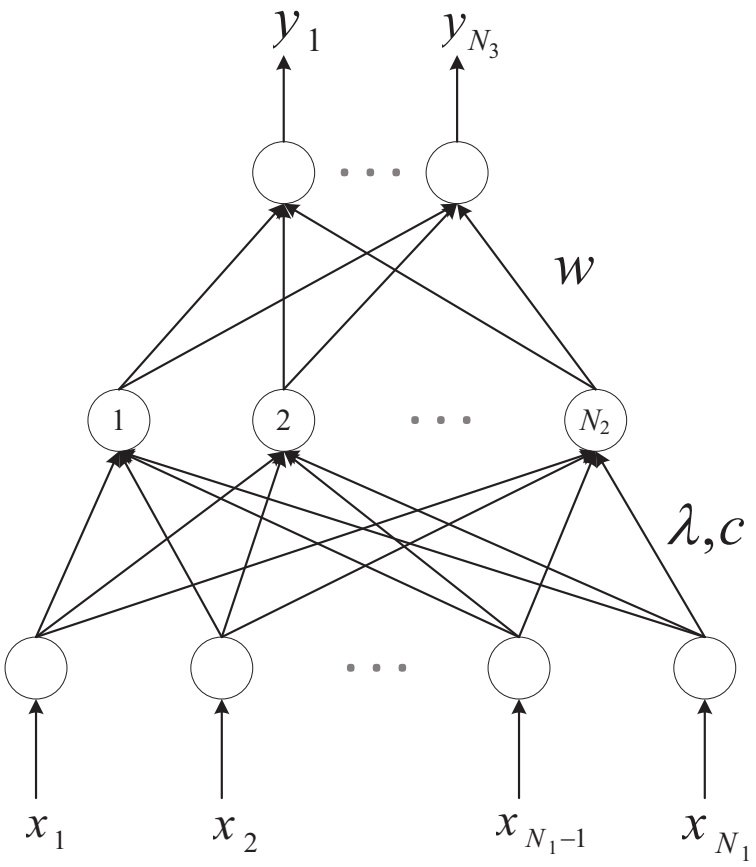


Figure 2.2: The structure of Radial basis function (RBF) neural network.

feedforward neural network structure such as MLP, RBF, and wavelet networks. The feedforward network together with the time delay and feedback mechanisms results in a recurrent neural network structure.

Another specific type of nerual network is called dynamic neural network (DNN). Dynamic neural network is described for large siganal modeling of nonlinear bahavior of the circuits in the circuit simulation. The stucture for dynamic neural network is illustrated in Fig. 2.4.

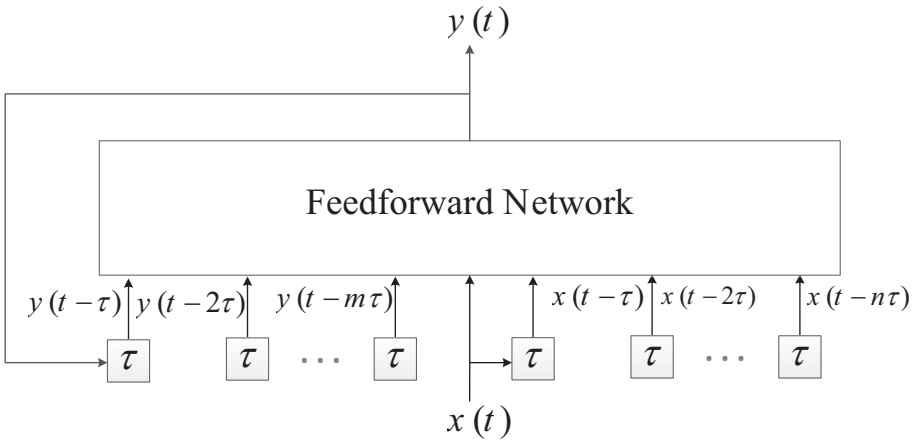


Figure 2.3: A recurrent neural network structure with feedback of delayed neural network output.

2.1.2 Neural network Training

A neural network cannot represent any device/circuit behavior unless it is well trained with corresponding data. A systematic neural model development process is described including data generation, data scaling, training, validation, and testing.

Let $\boldsymbol{x}$ represent an n_x-vector containing the design parameters of a RF/microwave component, called as input vector. Let $\boldsymbol{y}$ represent an n_y-vector containing the EM/electrical behavior of the component. The theoretical relationship between $\boldsymbol{x}$ and $\boldsymbol{y}$ can be represent by a neural network function, formulated as

$$\boldsymbol{y} = \boldsymbol{f}_{ANN}(\boldsymbol{x}, \boldsymbol{w}) \tag{2.1}$$

where $\boldsymbol{w}$ represents a vector containing all neural network weights. A fast and accurate neural model is developed by training the neural network function $\boldsymbol{f}_{ANN}$ with a set of simulated/measured data called the training data. The training data

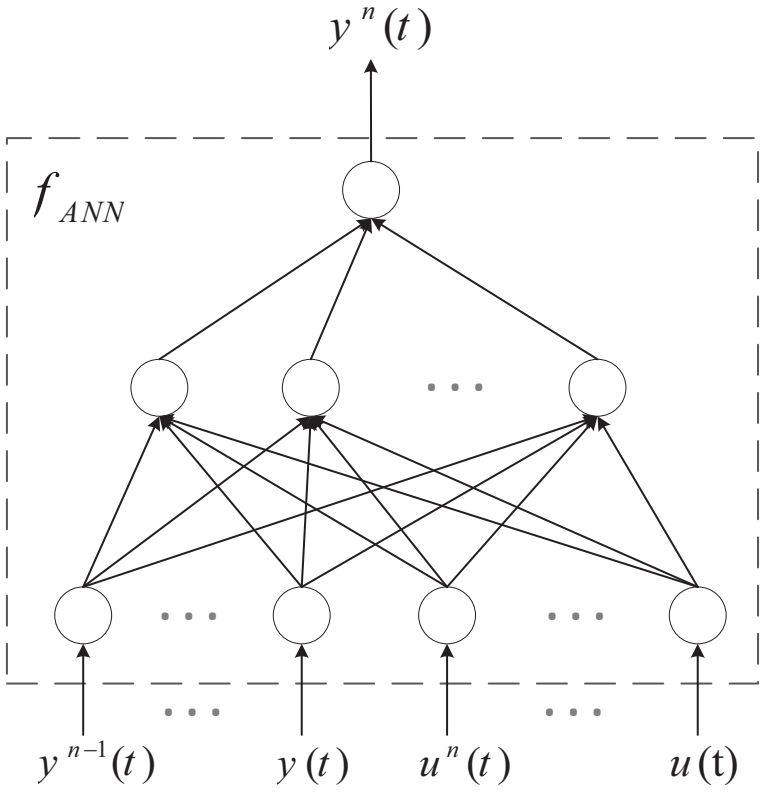


Figure 2.4: The structure of dynamic neural network.

is denoted by input-output sample pairs $\{(\boldsymbol{x}_k, \boldsymbol{d}_k), k \in T_r\}$, where $\boldsymbol{d}_k$ represents the measured/simulated output data for the input $\boldsymbol{x}_k$, and T_r represents the index set of training data.

For training purpose, the error function $E(\boldsymbol{w})$ is formulated as,

$$E(\boldsymbol{w}) = \frac{1}{2} \sum_{k \in T_r} \| \boldsymbol{f}_{ANN}(\boldsymbol{x}_k, \boldsymbol{w}) - \boldsymbol{d}_k) \|^2 \tag{2.2}$$

The primary objective of neural network training is to minimize $E(\boldsymbol{w})$ by adjusting the weigh parameters $\boldsymbol{w}$. Multiple training algorithms are developed for

RF/microwave neural modeling such as backpropagation, conjugate gradient, Quasi-Newton, Levenberg-Marquardt, Genetic Algorithms and Simulated Annealing. Once trained, an independent set of data, called test data, is used to test the accuracy of the neural network model. Normally, the test data should be generated within the same input range as the training data but contains the input-output samples which are never used in training stage. If both training and testing results are satisfied with the use desired accuracy, the neural network model are well trained and can be used in circuit simulation and design optimization.

2.2 Knowledge-Based Neural Network

The knowledge-based neural network (KBNN) is a modeling method combining prior knowledge with the learning ability of neural networks by incorporating the empirical information into the internal structure of neural networks. Fig. 2.5 illustrates the comprehensive structure of the knowledge-based neural network.

In KBNN, the microwave knowledge is embedded as part of the neural network internal structure. The KBNN structure includes six layers that are not fully connected to each other, called the input layer, the knowledge layer, the boundary layer, the region layer, the normalized region layer, and the output layer. The knowledge layer contains the microwave knowledge, complementing the ability of learning and generalization of neural networks with additional knowledge. The boundary layer incorporates knowledge in the form of problem dependent boundary functions. The region layer contains neurons for constructing regions from boundary neurons. The normalized region layer contains rational function based neurons to normalize the

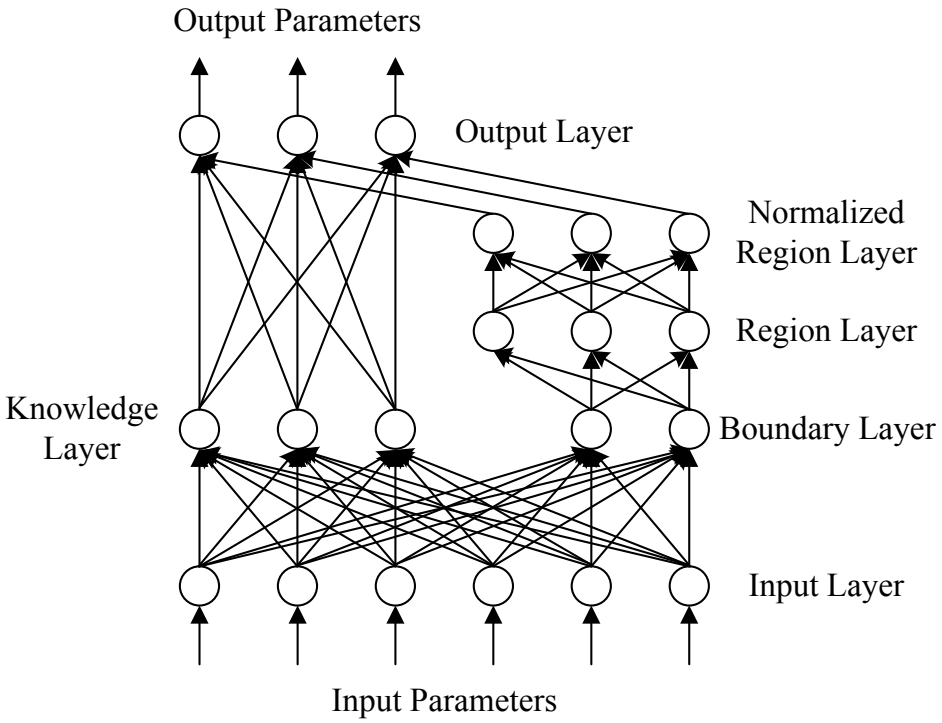


Figure 2.5: Illustration of the structure of knowledge-based neural network (KBNN) [35].

outputs of the region layer. The output layer consists of second-order neurons combining normalized region neurons and knowledge neurons.

The KBNN exploits existing knowledge in the form of empirical functions or equivalent circuit model together with a neural network model to develop faster and more accurate models. Extrapolation capability of KBNN is also enhanced by embedding knowledge in the model. Another parametric modeling and optimization technique with prior knowledge, using an advanced optimization concept named space mapping, is discussed in next section.

2.3 Space Mapping

Space mapping (SM) is a recognized engineering optimization methodology in the microwave area [19], [20]. The space mapping concept combines the computational efficiency of coarse models with the accuracy of fine models [19]. The coarse models are typically empirical functions or equivalent circuit models, which are computationally efficient. However, such models are often valid only in a limited region of input space, beyond which the model predictions become inaccurate. On the other hand, detailed or fine models can be provided by an electromagnetic (EM) simulator, or even by direct measurements. The detailed models are accurate, but can be expensive (e.g., CPU-intensive simulations). The space mapping technique establishes a mathematical link between the coarse and the fine models and directs the bulk of the CPU-intensive computations to the coarse model, while preserving the accuracy offered by the fine model [19]. Fig. 2.6 shows the illustration of the space mapping concept. Typically, space mapping algorithms provide excellent results after a few evaluations of the fine model.

Efforts on space mapping have focused on several areas, such as implicit space mapping [86], [87], output space mapping [88]−[90], neural space mapping [91]−[95], generalized space mapping [96], tuning space mapping [97], [98], portable space mapping [99], parallel space mapping [25], coarse and fine mesh space mapping [100], [101]. Recent improvements in space mapping such as constrained parameter extraction using implicit space mapping [102], space mapping optimization using EM-based adjoint sensitivity [103], and fast EM modeling using shape-preserving

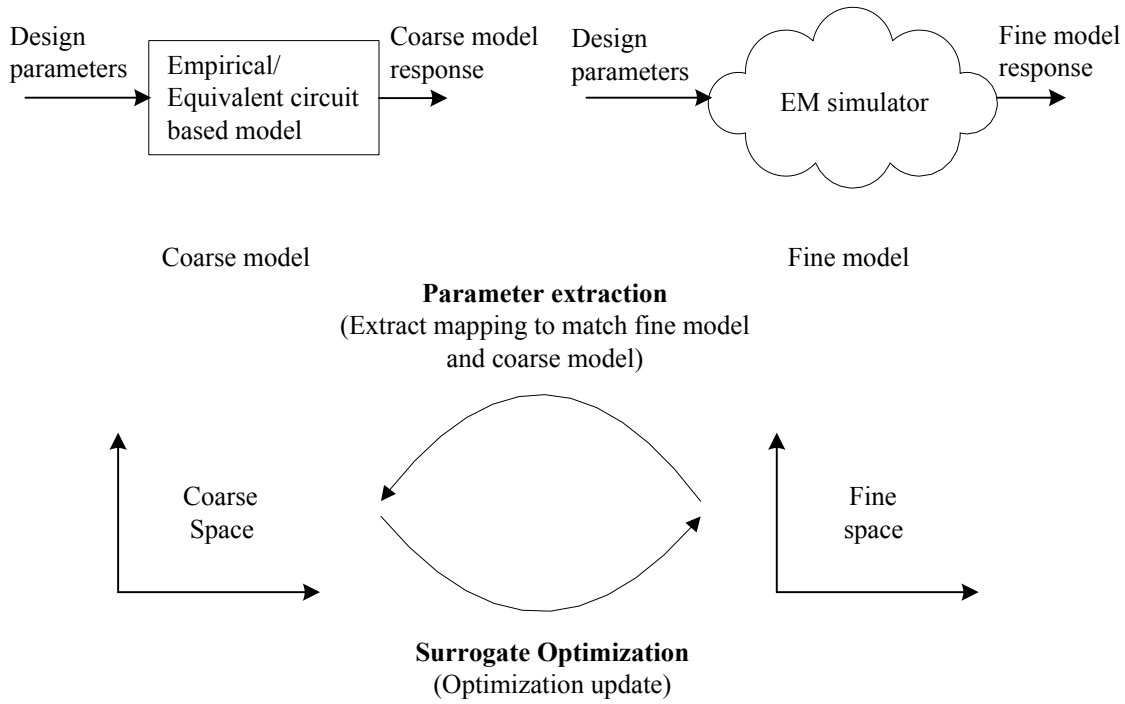


Figure 2.6: The illustration of the space mapping concept.

response prediction and space mapping [104] focus on reducing the number of fine model evaluations.

2.3.1 Space Mapping Concept

Let $\boldsymbol{R}_f(\boldsymbol{x}_f)$ denotes the response vector of a fine model corresponding to a vector of design variables $\boldsymbol{x}_f$. The original optimization problem is formulated as follows

$$\boldsymbol{x}_f^* = \arg\min_{\boldsymbol{x}_f} U(\boldsymbol{R}_f(\boldsymbol{x}_f)) \tag{2.3}$$

where U is a suitable objective function, which represents the error function of $\boldsymbol{R}_f(\boldsymbol{x}_f)$ with respect to the design specifications; $\boldsymbol{x}_f^*$ is the optimal fine model de-

sign to be found. We assume that solving problem (2.3) by means of direct EM optimization is computationally expensive. Instead, we exploit an inexpensive surrogate model; e.g., we establish a surrogate model which combines the coarse model with the input mapping function as

$$\boldsymbol{x}_c = \boldsymbol{P}_{SM}(\boldsymbol{x}_f) \tag{2.4}$$

such that

$$\boldsymbol{R}_s(\boldsymbol{x}_f) = \boldsymbol{R}_c(\boldsymbol{x}_c) = \boldsymbol{R}_c(\boldsymbol{P}_{SM}(\boldsymbol{x}_f)) \tag{2.5}$$

where $\boldsymbol{x}_c$ is a vector of design variables of the coarse model and $\boldsymbol{R}_c(\boldsymbol{x}_c)$ represents the response vector of the coarse model corresponding to $\boldsymbol{x}_c$. $\boldsymbol{R}_s(\boldsymbol{x}_f)$ is a response vector of the surrogate model and $\boldsymbol{x}_f$ is a vector containing all the design optimization variables. $\boldsymbol{P}_{SM}$ represents the space mapping function. The surrogate is trained to be very close to the fine model as

$$\boldsymbol{R}_c(\boldsymbol{P}_{SM}(\boldsymbol{x}_f)) \approx \boldsymbol{R}_f(\boldsymbol{x}_f) \tag{2.6}$$

Thus, the design optimization using surrogate model can represent that using fine model described in (2.3). The optimal solution of the surrogate model is denoted as

$$\boldsymbol{x}_f^* = \arg\min_{\boldsymbol{x}_f} U(\boldsymbol{R}_c(\boldsymbol{P}_{SM}(\boldsymbol{x}_f))) \tag{2.7}$$

Fig. 2.7 shows the mathematical representation of the space mapping methodology presented in the reference [19].

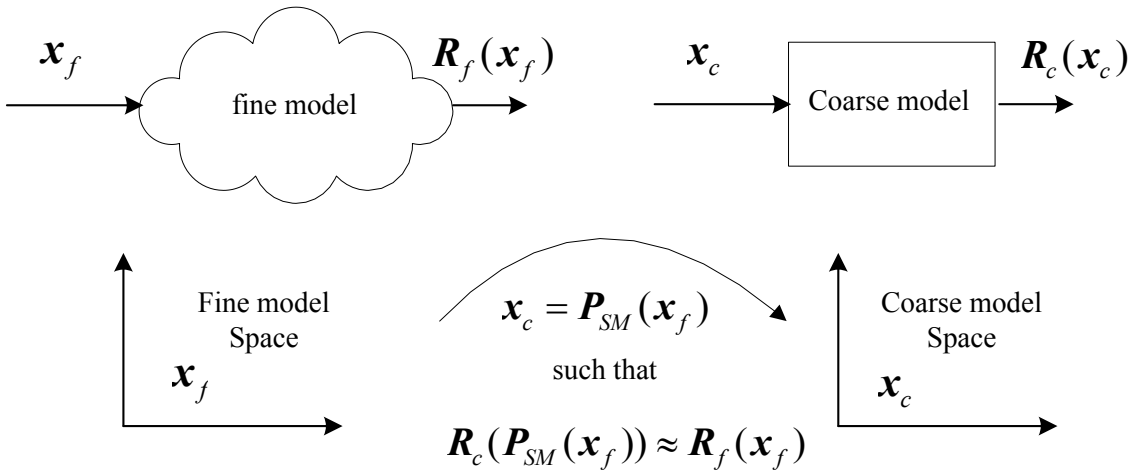


Figure 2.7: The mathematical representation of the space mapping methodology.

2.3.2 Input Space Mapping

At the beginning of the space mapping concept is presented, input space mapping is introduced as the most standard space mapping methodology [19]. Input space mapping focus on reducing the misalignment between the fine and coarse models by establishing a mapping between the input spaces (e.g., design parameter spaces) of the fine and coarse models. Input mapping with the linear mapping function is also called as original space mapping, defined as

$$\boldsymbol{x}_c = \boldsymbol{P}_{SM}(\boldsymbol{x}_f) = \boldsymbol{B}_{SM}\boldsymbol{x}_f + \boldsymbol{c}_{SM} \tag{2.8}$$

where $\boldsymbol{B}_{SM}$ and $\boldsymbol{c}_{SM}$ represent the coefficients of a linear mapping function.

2.3.3 Implicit Space Mapping

Implicit space mapping [86], [87] explores the flexibility of the preassigned parameter such as dielectric constant, substrate height in the design optimization process. Selected preassigned parameters (e.g., dielectric constant and substrate height) are extracted to match the coarse and fine models. The idea of using preassigned parameters was introduced in the reference [87] within an expanded space mapping design framework. This method selects certain key preassigned parameters based on sensitivity analysis of the coarse model. These parameters are extracted to match corresponding coarse and fine models. A mapping from optimization parameters to preassigned parameters is then established. Let $\boldsymbol{x}_{aux}$ represent the auxiliary parameters (i.e., preassigned parameters) and $\boldsymbol{R}_c(\boldsymbol{x}_c, \boldsymbol{x}_{aux})$ represent the coarse model response. As illustrated in Fig. 2.8, implicit space mapping aims at establishing an implicit mapping $\boldsymbol{Q}_{SM}$ between the spaces $\boldsymbol{x}_f$, $\boldsymbol{x}_c$, and $\boldsymbol{x}_{aux}$,

$$\boldsymbol{Q}_{SM}(\boldsymbol{x}_f, \boldsymbol{x}_c, \boldsymbol{x}_{aux}) = 0 \tag{2.9}$$

such that

$$\boldsymbol{R}_f(\boldsymbol{x}_f) \approx \boldsymbol{R}_c(\boldsymbol{x}_c, \boldsymbol{x}_{aux}) \tag{2.10}$$

Implicit mapping produces a good match between the coarse and fine models in the first iteration when input space mapping alone cannot obtain a good match.

2.3.4 Output Space Mapping

It is important for the performance of space mapping that the surrogate model response can represent the fine model response well with a proper input or implicit

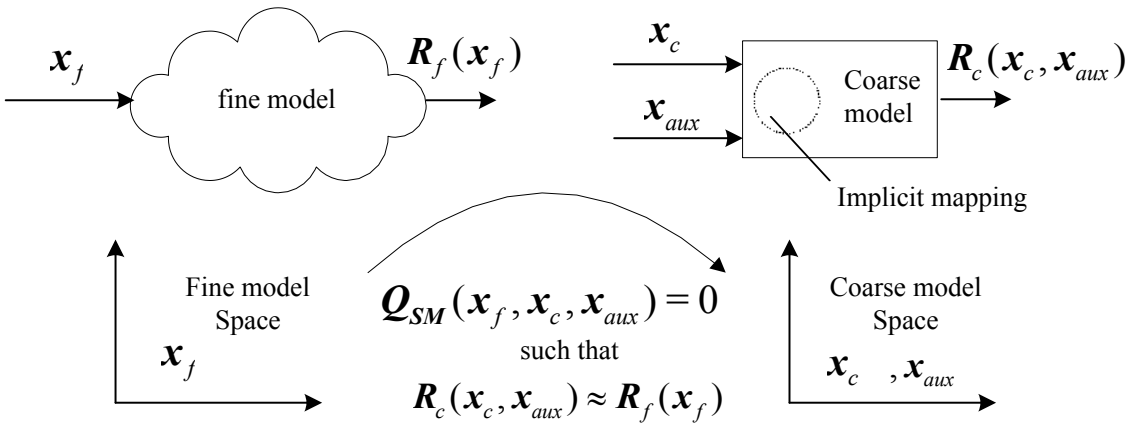


Figure 2.8: The mathematical representation of the space mapping methodology.

mapping. However, with only input or implicit space mapping, the surrogate model response may not be enough to precisely represent the fine model response. Therefore, a so-called output space mapping has been introduced [96], [105]. Output space mapping enhances the surrogate model by a correction term which is the difference between the fine model and the original space mapping response at the current iteration, [106]−[108], formulated as,

$$\boldsymbol{R}_s(\boldsymbol{x}_f) = \boldsymbol{P}_{corr}(\boldsymbol{x}_f) + \boldsymbol{R}_c(\boldsymbol{P}_{SM}(\boldsymbol{x}_f)) \tag{2.11}$$

where $\boldsymbol{P}_{corr}(\boldsymbol{x}_f)$ represents the added correction term at the output space in surrogate model.

The surrogate model is further enhanced by using the Jacobin of the surrogate model (to satisfy first order consistency between the surrogate model and fine model at the current design). If the misalignment between the fine and coarse models is not significant, SM-based optimization algorithms typically provide excellent results

after only a few evaluations of the fine model. Having a large number of space mapping types results in a even larger number of combinations for space mapping. So the need to choose the suitable coarse model and SM approach for a design problem is crucial. A suitable choice of SM approach requires both knowledge of the problem and engineering experience.

2.3.5 Tuning Space Mapping

Tuning space mapping (TSM) [98] is a special type of SM technique that caters to tuning of EM structures. The surrogate model is replaced by a tuning model which introduces circuit components in to the fine model structure. The tuning model is optimized within a circuit simulator. With the optimal tuning parameters, thus obtained, they are mapped or transformed into the design variables using fast space-mapping surrogate or analytical formulas if available. Tuning models require a significant engineering expertise for a successful implementation of the optimization process using TSM approach.

2.3.6 Neuro Space Mapping

The most frequently used space mapping technique is developed to use linear mappings to establish a mathematical link between the coarse model and the fine data. However, when the modeling range becomes large, linear mappings only are not enough. Neuro space mapping is presented to solve this problem [17]. The neural networks are used to provide a nonlinear computational approach to bridge the gap between the empirical/equivalent circuit model and the new EM simulation data.

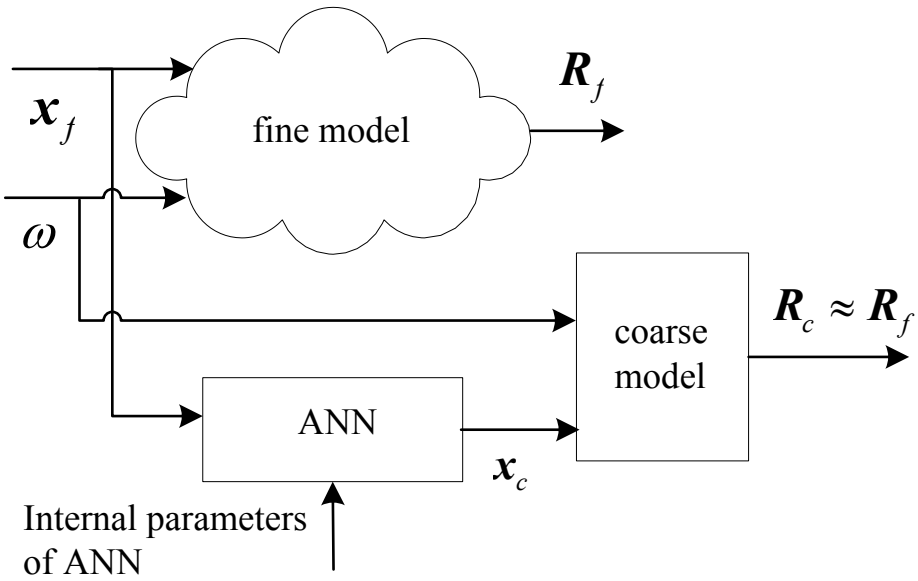


Figure 2.9: The mathematical representation of the space mapping methodology.

This is achieved with the space mapping concept using neural networks to represent the nonlinear mappings between the empirical/equivalent circuit model and the EM data. Fig. 2.9 illustrate the neuro space mapping concept. Extrapolation capability is also enhanced because of the embedded knowledge in the model [35]. Several approaches for the structure selection of the neuro space mapping model are described in the existing literature [91]−[95].

2.3.7 Parallel Space Mapping

Parallel computation is a powerful method to speed up intensive computational processes and utilize computer's number crunching ability more effectively [109]. Many researches on parallel method have been done in several areas [110]−[117]. Parallel automatic model generation technique is proposed in the reference [110], using par-

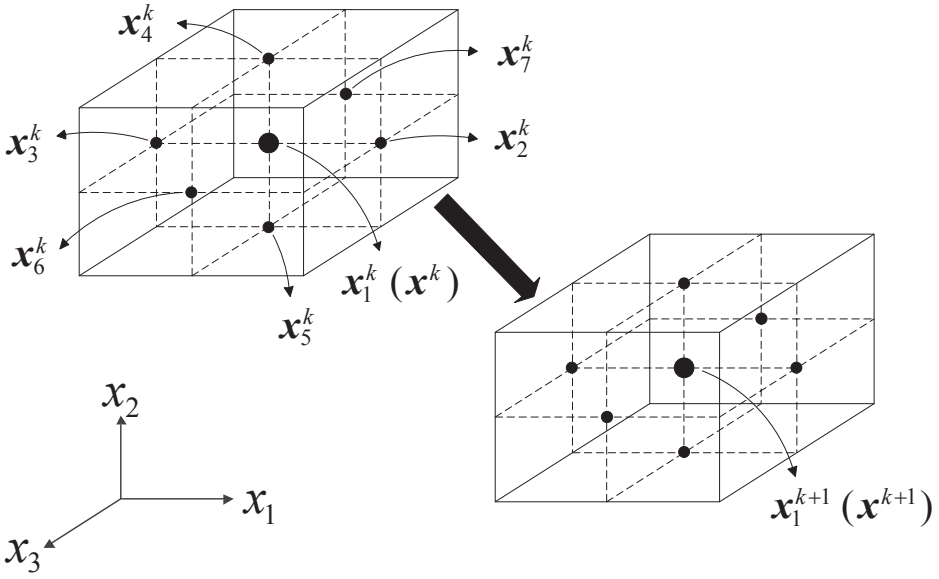


Figure 2.10: Illustration of multiple fine model points used to train the surrogate model.

allel adaptive sampling and parallel data generation to save the model development time. In the reference [115], the EM data is generated by running multiple EM simulations in parallel on a multi-processor environment. This technique is used to speed up the design optimization of microwave circuits. In the references [116], [117], distributed fine model evaluation technique has been presented.

In the reference [25], a parallel space mapping optimization algorithm is presented. The surrogate model developed in each iteration is trained to match the fine model at multiple points, thereby making the surrogate model to be valid in a larger neighborhood. The formulation of multi-point surrogate model training is inherently suited to and implemented through parallel computation. This includes multiple fine model evaluation in parallel and multi-point surrogate training using

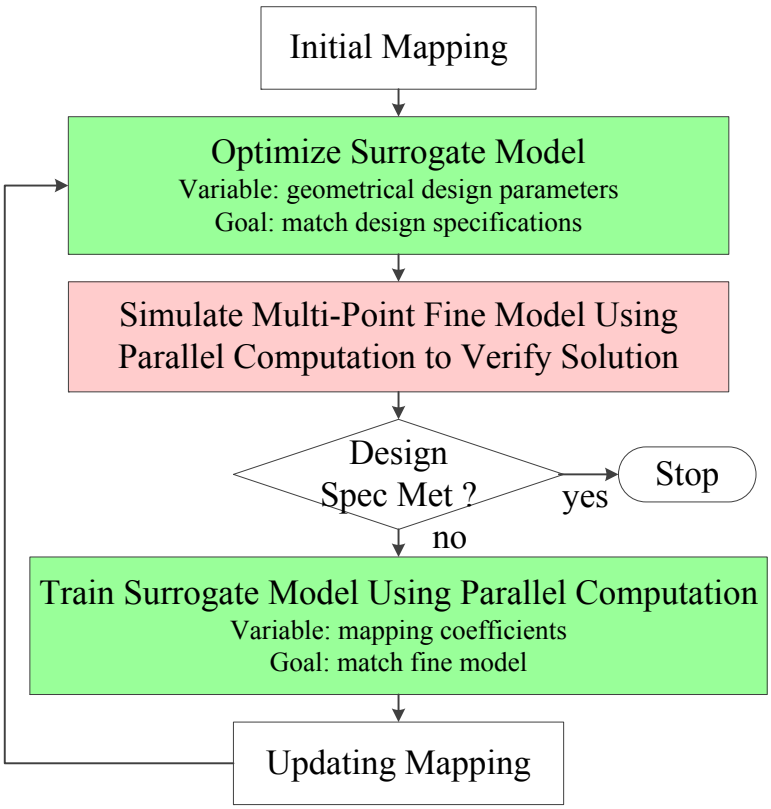


Figure 2.11: The flowchart of the parallel space mapping algorithm.

parallel algorithm. Fig. 2.10 illustrate the multiple fine model points used to train the surrogate model. Fig. 2.11 shows the flowchart of the parallel space mapping algorithm. The parallel space mapping further reduces the number of space mapping iterations and speeds up the optimization process.

The standard space mapping technique needs prior knowledge, called as coarse model, in order to perform parametric modeling and design optimization. Recent advanced parametric modeling and optimization techniques with no coarse model needed are reviewed in next section.

2.4 Overview of Parametric Modeling and Optimization with No Available Coarse Model

In recent days, EM simulations more focus on 3D and complex structures, whose equivalent circuits or empirical equations are difficult to formulate. Under that situations, developing parametric modeling techniques with no available coarse model becomes a popular research direction. Here discusses an overview of several parametric modeling and optimization techniques without available coarse model.

2.4.1 Coarse and Fine Mesh Space Mapping

The space mapping normally requires an equivalent circuit as the coarse model. The coarse and fine mesh space mapping addresses the situation when an equivalent circuit coarse model is not available [100], [101]. In the coarse and fine mesh space mapping, fine model uses a fine mesh for EM simulation. Fine model simulation and mesh refinement are performed iteratively until the simulation results between successive iterations converge. Fine mesh simulation is accurate but computationally expensive. On the other hand, coarse model uses a coarse mesh for EM simulation, i.e., local mesh without mesh convergence. Also, a coarse model is computationally fast but less accurate. A simple illustration of coarse mesh and fine mesh with a microstrip bandstop filter example is shown in Fig. 2.12. The coarse and fine mesh space mapping uses mostly coarse mesh EM evaluation and occasionally fine mesh EM evaluation to achieve optimal EM solutions with fine mesh accuracy.

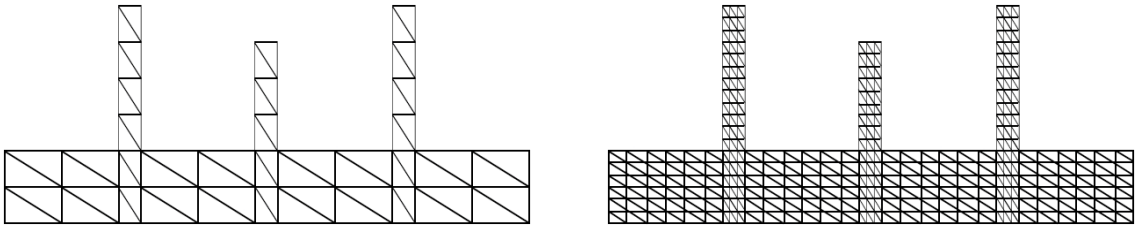


Figure 2.12: The illustration of coarse mesh and fine mesh with a microstrip band-stop filter example.

2.4.2 Cognition-Driven Formulation of Space Mapping for Equal-Ripple Optimization of Microwave Filters

Existing space-mapping approaches belong to the class of surrogate-based optimization methods. In the reference [118], a cognition-driven formulation of space mapping that does not require explicit surrogates is discussed. This method is applied to EM-based filter optimization. This technique utilizes two sets of intermediate feature space parameters, including feature frequency parameters and ripple height parameters. The design variables are mapped to the feature frequency parameters, which are further mapped to the ripple height parameters. By formulating the cognition-driven optimization directly in the feature space, this method increases optimization efficiency and the ability to avoid being trapped in local minima. The technique is suitable for design of filters with equal-ripple responses.

2.4.3 Parallel EM Optimization using Transfer Function-based Surrogate Model

In existing EM optimization, surrogate models are constructed using single point fine model data, or accumulated fine model data from previous iterations [19], [20]. In the reference [25], multiple samples using star distribution are used for constructing surrogate model. The availability of a fast coarse model or problem-dependent prior knowledge is a pre-requisite for these techniques. In reality, not all designs can satisfy this pre-requisite.

The method in the reference [119] aims to address EM optimization without the use of coarse models. In this way, the proposed technique is useful for EM optimization even when the empirical or equivalent circuits models for the EM structure are not available. In the reference [119], a widely accepted transfer functions to represent the EM behavior for constructing the surrogate. The surrogate model response is expressed as a transfer function in the rational function format. A further research [120] uses decomposition technique for parallel EM optimization with transfer function-based surrogate model. This method can handle more geometrical parameters as design variables by using the decomposition technique while performing the EM optimization.

2.4.4 Parametric Modeling using Combined Neural Networks and Transfer Functions

Recently, advanced modeling approach, which combines neural networks and transfer functions (neuro-TF), was developed to perform parametric modeling of EM

responses [26], [28], [29]. This approach can be used even if accurate equivalent circuits or empirical models are unavailable. In this method, transfer functions are used to represent the EM responses of passive components versus frequency.

The model consists of transfer functions and neural networks. The outputs of the overall model are the S-parameters of the EM behavior of microwave components and the inputs of the model are geometrical variables of the EM structure and frequency. As the values of geometrical parameters change, the coefficients of transfer functions change accordingly. Due to the relationship between the coefficients of transfer functions and the geometrical parameters is nonlinear and unknown, neural networks are used to learn and represent this nonlinear relationship. The initial training data of neural networks are obtained by the vector fitting technique [121]. With vector fitting, the coefficients of transfer functions corresponding to a given set of EM responses are obtained. The neural networks are trained to learn the nonlinear mapping between geometrical parameters and the coefficients of transfer functions.

2.5 Conclusion

In this chapter, a literature review of popularly used EM parametric modeling and design optimization techniques has been discussed. An overview of artificial neural network methods has been presented. The structure of artificial neural network and its training methods have been reviewed. The knowledge-based neural network has also been discussed as an advanced type of neural networks with prior knowledge as part of internal neural network structure. Another knowledge-based technique

called space mapping methods such as input space mapping, implicit space mapping, output space mapping, tuning space mapping, neural space mapping, and recently developed parallel space mapping has also been presented. Further, recent advanced EM parametric modeling and optimization methods addressing the situation with no available coarse model have also been reviewed. In the next chapter, a novel parametric modeling of EM behavior of microwave components using combined neural networks and pole-residue-based transfer functions is proposed.

Chapter 3

Parametric Modeling Using Pole-Residue-Based Neuro-Transfer Functions

This chapter describes a novel and advanced technique to develop combined neural network and pole-residue-based transfer function models for parametric modeling of electromagnetic (EM) behavior of microwave components. In this technique, neural networks are trained to learn the relationship between pole/residues of the transfer functions and geometrical parameters. The order of the pole-residue transfer function may vary over different regions of geometrical parameters. We develop a pole-residue tracking technique to solve this order-changing problem. After the proposed modeling process, the trained model can be used to provide accurate and fast prediction of the EM behavior of microwave components with geometrical parameters as variables. The proposed method can obtain better accuracy in challenging applications involving high dimension of geometrical parameter space and large geometrical variations, compared with conventional modeling methods. The proposed

technique is effective and robust especially in solving high-order problems. This technique is illustrated by three examples of EM parametric modeling.

3.1 Introduction

In the field of parametric modeling for microwave components, an advanced modeling approach, which combines neural networks and transfer functions (neuro-TF), was developed to perform parametric modeling of EM responses [26], [27]. This approach can be used even if accurate equivalent circuits or empirical models are unavailable. In this method, transfer functions are used to represent the EM responses of passive components versus frequency. However, the coefficients of transfer functions have discontinuity problems when there are large geometrical variations. In the reference [28], a training method for developing parametric neuro-TF model of microwave passive components in the frequency domain has been presented under the rational transfer function formulation. The solution of the discontinuity of coefficients in transfer functions over the geometrical variables is addressed in the reference [28]. The sensitivities of the coefficients of the rational transfer function with respect to geometrical parameters become high when the order of transfer function is high and/or the geometrical variations are large. This limits the accuracy and robustness of the model. Reference [29] presented an alternative pole-residue approach to the transfer function. However, the method to address discontinuity in the reference [28] applicable to rational functions is not applicable to pole-residue-based transfer functions, which requires a quite different approach. The order-changing problem for the pole/residue formulation remains an open problem when the geo-

metrical variations are large.

In this chapter, a novel technique is proposed to develop a combined neural network and pole-residue-based transfer function (pole-residue-based neuro-TF) model for parametric modeling of EM behavior of microwave components. A new method is proposed to address the discontinuity issue of the pole-residue-based transfer function. In our method, a novel order-changing technique is developed for pole-residue-based transfer functions to handle the problem of order variations as geometrical parameters change in a large region. With high dimension of geometrical parameter space and large geometrical variations, our proposed method produces more accurate models than the existing order-changing method using rational transfer function in the reference [28], exploiting the advantage that pole/residues have low sensitivities w.r.t geometrical parameters than rational function coefficients do. The accuracy of the model developed by the proposed method remains good even when the order of the model is high. After the proposed modeling process, the trained model can be used to provide accurate and fast prediction of the EM response with respect to the geometrical variables and can be subsequently used in high-level circuit and system design. The proposed technique is effective and robust especially in solving high-order problems.

3.2 Proposed Technique for Developing a Pole-Residue-Based Neuro-TF Model

3.2.1 Formulation of The Pole-Residue-Based Neuro-TF Model

The structure of the proposed pole-residue-based neuro-TF model is illustrated in Fig. 3.1. The model consists of pole-residue-based transfer functions and neural networks. The outputs of the overall model are the S-parameters of the EM behavior of microwave components and the inputs of the model are geometrical variables of the EM structure and frequency.

Let $\boldsymbol{x}$ be a vector containing the geometrical variables, representing the inputs of the overall model. Let the frequency response $H(s)$ be a function of pole/residues, which is defined using a pole-residue-based transfer function as follows,

$$H(s) = \sum_{i=1}^{N} \frac{r_i}{s - p_i} \tag{3.1}$$

where p_i and r_i represent the poles and residues of the transfer function respectively, and N represents the order of the transfer function.

As the values of geometrical parameters change, the pole/residues change accordingly. Due to the relationship between the pole/residues and the geometrical parameters is nonlinear and unknown, we propose to use neural networks to learn and represent this nonlinear relationship. The three layer MLP is used as the neural network structure with linear functions as the output layer activation functions and sigmoid functions as the hidden layer activation functions. The initial training data of neural networks are obtained by the vector fitting technique [121]. With vector

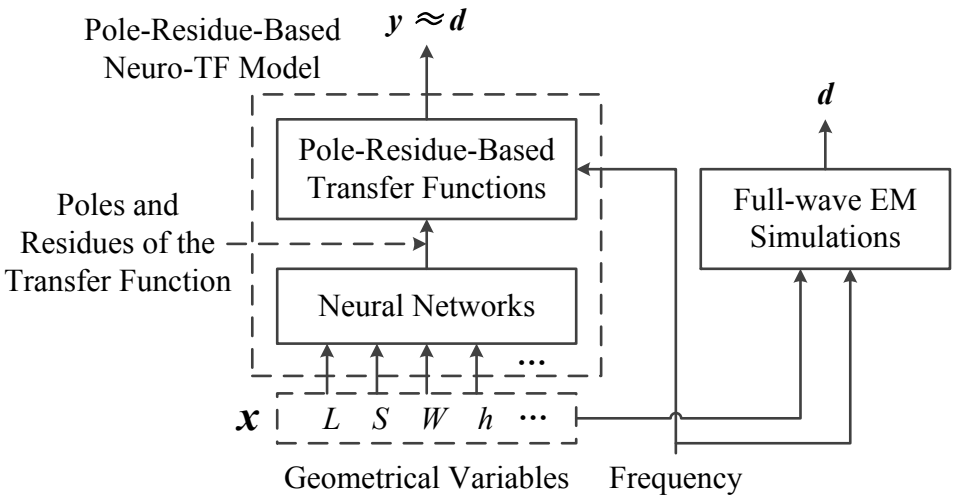


Figure 3.1: The structure of the pole-residue-based neuro-TF model. $\boldsymbol{x}$ represents the geometrical variables. $\boldsymbol{y}$ represents real and imaginary parts of the outputs of the pole-residue-based transfer function (e.g., S-parameters). $\boldsymbol{d}$ represents the outputs of the EM simulations.

fitting, we obtain the poles and residues of the transfer function corresponding to a given set of EM responses. The neural networks are trained to learn the nonlinear mapping between $\boldsymbol{x}$ and the pole/residues. Let $\boldsymbol{y}$ be a vector representing real and imaginary parts of the outputs of the pole-residue-based transfer function. Let $\boldsymbol{d}$ be a vector representing the outputs of the EM simulations (e.g., real and imaginary parts of S-parameters). The objective here is to minimize the error between $\boldsymbol{y}$ and $\boldsymbol{d}$ for different $\boldsymbol{x}$, by adjusting the neural network internal weights.

We use the vector fitting process [121] to extract the coefficients (or pole/residues) of the transfer function. When the frequency range is large with respect to the central frequency point, higher order transfer function needs to be used in order to obtain a good vector fitting accuracy. However, high order transfer function results

in non-uniqueness and discontinuity problems of the coefficients of the rational formulation (or pole/residues) of the transfer functions. In order to perform vector fitting to obtain accurate results with the minimum orders of the transfer functions, we scale and shift the frequency range to a small range relatively far away from the zero frequency. As illustrated in Table 3.1, for the rational transfer function, when the frequency range is small (e.g., varying around 1 to ignore the influence of s^j), all the poles p_i are close to each other. When the value of frequency s is in the middle of all the p_i, the value of $\prod_{i=1}^{N}(s-p_i)$ becomes very small, which makes the $H(s)$ too sensitive with respect to the coefficients a_j and b_j of the rational transfer function. On the other hand, the sensitivity of the $H(s)$ with respect to all the poles and residues remains low no matter how large the order of the transfer function becomes. This feature of the pole-residue-based transfer function makes the model more robust than the rational transfer function formulation when dealing with high order modeling problems. In this work, we use frequency scaling and shifting to achieve the minimum order, and use poles and residues as intermediate variables instead of the coefficients of the rational transfer function of [26],[28].

3.2.2 The Issue of Discontinuity in Poles and Residues with respect to Geometrical Parameters

The proposed technique for developing a pole-residue-based neuro-TF model begins with samples of EM data ($\boldsymbol{x}_k$, $\boldsymbol{d}_k$) for different values of geometrical parameters, where the subscript k is the index indicating the k^{th} sample of geometrical parameters, $k \in T_r = \{1, 2, ..., n_s\}$. T_r is the index set of training samples of geometrical

Table 3.1: Comparisons of Different Sensitivities of Transfer Functions

	Coefficient (or Pole/Residue)	Transfer Function	Sensitivity of Transfer Function w.r.t. Coefficient (or Pole/Residue)
Rational Transfer Function	a_j	$H(s)=\frac{\sum_{i=0}^{M} a_i s^i}{1+\sum_{i=1}^{N} b_i s^i}$	$\frac{\partial H}{\partial a_j}=\frac{s^j}{1+\sum_{i=1}^{N} b_i s^i}=\frac{s^j}{\prod_{i=1}^{N}(s-p_i)}$
	b_j		$\frac{\partial H}{\partial b_j}=\frac{-s_j\sum_{i=0}^{M} a_i s^i}{\left(1+\sum_{i=1}^{N} b_i s^i\right)^2}=H\cdot\frac{-s^j}{\prod_{i=1}^{N}(s-p_i)}$
Pole-Residue-Based Transfer Function	r_j	$H(s)=\sum_{i=1}^{N}\frac{r_i}{s-p_i}$	$\frac{\partial H}{\partial r_j}=\frac{1}{s-p_j}$
	p_j		$\frac{\partial H}{\partial p_j}=\frac{r_j}{(s-p_j)^2}$

parameters and n_s is the total number of training samples. Training samples are generated with respect to inputs $\boldsymbol{x}$ including the geometrical parameters except frequency which is a separate variable swept by the EM simulator during data generation. In our method, frequency is an additional input of the pole-residue-based transfer function. $\boldsymbol{d}$ represents real and imaginary parts of S-parameters generated from EM simulations.

The discontinuity of pole/residues in transfer functions with respect to the geometrical variables is a major issue for modeling. When geometrical variations are large, the corresponding EM responses will lead to different orders of transfer functions. A repetitive parameter extraction process is required to extract the pole-residues of transfer functions for each sample of geometrical variables. One way is to set the orders to the maximum one among all geometrical samples. In that case, high order transfer function is used to deal with low order problems for subsets of geometrical samples. That causes non-unique and arbitrary numerical solutions for

the vector fitting process, resulting in the discontinuity of pole/residues.

Another way is to use a minimum order of the pole-residue-based transfer function for each geometrical sample [29]. That may result in different orders of transfer functions for different geometrical samples. These varied orders will result in abrupt changes of the values of poles and residues, and/or abrupt changes in the function structure of poles and residues as functions of geometrical parameters. These abrupt changes also lead to the discontinuity of pole/residues. Table 3.2 illustrates an example of this discontinuity problem of pole/residues occurring when a new pole is added resulting in the change of the orders. Assume that when the length (L) and width (W) (i.e., geometrical parameters) of the components change across the boundary from $[L_1\ W_1]^T$ to $[L_1+\Delta L\ W_1+\Delta W]^T$, causing the order to change from two to three, a new pole $p_1+\Delta p_1'$ is split from the first pole. By using the brute-force minimum order method, the values of $H_1(s)$ and $H_2(s)$ are the same, when ΔL and ΔW approach to zeros. However, the pole/residues $\{P_1\ P_2\ P_3\ R_1\ R_2\ R_3\}$ in $H_1(s)$ and $H_2(s)$ do not approach each other. In other words, the limits of pole/residues of $H_2(s)$ and $H_1(s)$ are quite different from each other. In this case, the pole/residues obtained using the brute-force minimum order method in some columns of Table 3.2 are discontinuous (i.e., change abruptly) when the orders of transfer functions are changed over the variations of geometrical parameters. By similar analysis, the discontinuity problem will occur no matter which pole should be split. Such discontinuity problem cannot be solved by the order changing method in [28]. The order changing process in [28] works well for transfer function in rational format, since the newly added coefficients of transfer functions after order changing always corre-

Table 3.2: Discontinuity of Pole/Residues When The Orders of Transfer Functions are Changed

Parameters		Transfer Function	Poles and Residues					
L	W		P_1	P_2	P_3	R_1	R_2	R_3
L_1	W_1	$H_1(s)=\frac{r_1}{s-p_1}+\frac{r_2}{s-p_2}$	p_1	p_2	0	r_1	r_2	0
$L_1+\Delta L$	$W_1+\Delta W$	$H_2(s)=\frac{r_1/2+\Delta r_1}{s-(p_1+\Delta p_1)}+\frac{r_1/2+\Delta r_1'}{s-(p_1+\Delta p_1')}+\frac{r_2+\Delta r_2}{s-(p_2+\Delta p_2)}$	$p_1+\Delta p_1$	$p_1+\Delta p_1'$	$p_2+\Delta p_2$	$\frac{r_1}{2}+\Delta r_1$	$\frac{r_1}{2}+\Delta r_1'$	$r_2+\Delta r_2$
$L_1+\Delta L$	$W_1+\Delta W$	$\lim_{\substack{\Delta r_1,\Delta r_1',\Delta r_2\to 0\\ \Delta p_1,\Delta p_1',\Delta p_2\to 0}} H_2(s)$	p_1	p_1	p_2	$\frac{r_1}{2}$	$\frac{r_1}{2}$	r_2

spond to the highest order terms in the transfer function. However, the sequence of pole/residues for a geometrical sample and the relationship of pole/residues among different geometrical samples cannot be easily decided using [28]. Thus we need to develop a new algorithm to track the poles for order changing of transfer functions in pole-residue format.

In subsequent subsections, we introduce a novel pole-residue tracking technique to overcome the discontinuity problems of pole/residues in transfer functions. The idea is based on splitting one pole into two separate new poles to achieve the increase of the order of the pole-residue-based transfer function. This new approach generates a group of new pole/residues based on those in the minimum order transfer functions and ultimately form transfer functions of constant order. The method solves the discontinuity problems of pole/residues while maintaining the accuracy of the transfer functions.

3.2.3 Vector Fitting for Parameter Extraction

Firstly, vector fitting process [121] is performed to obtain a group of poles $\boldsymbol{p}$ and residues $\boldsymbol{r}$, using the minimum order of the pole-residue-based transfer function for each geometrical sample. In the vector fitting process, the given information is EM data $\boldsymbol{d}$ (i.e., S-parameters) versus frequency for a geometrical sample. Expected solutions are poles and residues of the transfer function.

A scaling and shifting process for the frequency range is set up during the vector fitting process to get all poles and residues in complex values.

Let $\boldsymbol{c}_k$ represent a vector containing the poles and residues of the transfer function of the k^{th} geometrical sample obtained after vector fitting, defined as

$$\boldsymbol{c}_k = \begin{bmatrix} \boldsymbol{p}^{(k)} \\ \boldsymbol{r}^{(k)} \end{bmatrix} = [p_1^{(k)} p_2^{(k)} \cdots p_{n_k}^{(k)} \;\; r_1^{(k)} r_2^{(k)} \cdots r_{n_k}^{(k)}]^T \tag{3.2}$$

where n_k represents the order of the transfer function for the k^{th} geometrical sample. $\boldsymbol{p}^{(k)}$ is a vector of poles for the k^{th} geometrical sample, defined as $\boldsymbol{p}^{(k)} = [p_1^{(k)} p_2^{(k)} \cdots p_{n_k}^{(k)}]^T$. $\boldsymbol{r}^{(k)}$ is a vector of residues for the k^{th} geometrical sample, defined as $\boldsymbol{r}^{(k)} = [r_1^{(k)} r_2^{(k)} \cdots r_{n_k}^{(k)}]^T$.

We want to build a mapping function to map the pole/residues over geometrical variables. Because the mapping relationship is nonlinear and unknown, we use neural networks as the mapping function. Since there are different minimum orders for different geometrical values, the number of elements in vector $\boldsymbol{c}_k$ will be different for different geometrical sample. This variation of the numbers of pole/residues among different geometrical samples will cause difficulty in training the neural networks.

We proposed a novel pole-residue tracking technique to handle the change in the order of transfer functions.

3.2.4 Pole-Residue Tracking Technique for Order-Changing

In this section, a novel pole-residue tracking technique solving the problem due to the change in the order of transfer functions is proposed. The main purpose for this technique is to add groups of new pole/residues to bridge the gap of transfer function orders between different geometrical samples while keeping the responses of the transfer functions unchanged. In this way, we ultimately obtain the transfer functions of constant order with respect to all geometrical samples, overcoming non-uniqueness problem of pole/residues.

Vector fitting with frequency scaling and shifting is performed to obtain minimum order transfer function for each geometrical sample. Since all the poles and residues are complex numbers, the order of each sample, i.e., n_k should be an even number.

The first step of pole-residue tracking technique is to compact the poles and residues into the effective poles and residues. Based on the pole-residue-based transfer function in (3.1), each pole has a related residue. We will use complex pole/residues to formulate a systematic pole-residue tracking process. Since each complex pole or residue has a conjugate, we remove all of the poles which has negative imaginary part, and the corresponding residues. After that process, the number of pole/residues are reduced by half, and the remaining ones are regarded as effective pole/residues. We define the effective order N_k of the transfer function for the

k^{th} sample as

$$N_k = n_k/2 \tag{3.3}$$

The maximum and the minimum effective orders among all geometrical samples are defined as

$$N_{\max} = \max_{k \in T_r}\{N_k\} \tag{3.4}$$

$$N_{\min} = \min_{k \in T_r}\{N_k\} \tag{3.5}$$

where T_r is the index set of training samples for training the neural networks, defined as $T_r = \{1, 2, ..., n_s\}$, and n_s is the total number of the training samples of geometrical parameters. The effective poles include all the poles which have positive imaginary parts. The effective residues include all the residues corresponding to the effective poles.

The second step is to sort the effective pole/residues. Poles are sorted in an ascending sequence according to the values of their imaginary parts, while residues are sorted with their corresponding poles. Binary tree sorting method is used in this step. The average cost of this sorting algorithm is $N_k \log N_k$ for each geometrical sample, where N_k is the effective order of the transfer function for the k^{th} sample. The sorting process guarantees the relative correspondence of pole/residues between different samples (i.e., between different sets of geometrical values), therefore, ensuring a smooth and continuous model with respect to varying values of geometrical parameters.

The third step is the main pole-residue tracking process. Since the transfer functions created with the effective poles and residues have different effective orders

Table 3.3: Elimination of Discontinuity of Pole/Residues Using Pole-Residue Tracking Technique

Parameters		Transfer Function	Poles and Residues					
L	W		P_1	P_2	P_3	R_1	R_2	R_3
L_1	W_1	$H_1(s)=\frac{r_1/2}{s-p_1}+\frac{r_1/2}{s-p_1}+\frac{r_2}{s-p_2}$	p_1	p_1	p_2	$\frac{r_1}{2}$	$\frac{r_1}{2}$	r_2
$L_1+\Delta L$	$W_1+\Delta W$	$H_2(s)=\frac{r_1/2+\Delta r_1}{s-(p_1+\Delta p_1)}+\frac{r_1/2+\Delta r_1'}{s-(p_1+\Delta p_1')}+\frac{r_2+\Delta r_2}{s-(p_2+\Delta p_2)}$	$p_1+\Delta p_1$	$p_1+\Delta p_1'$	$p_2+\Delta p_2$	$\frac{r_1}{2}+\Delta r_1$	$\frac{r_1}{2}+\Delta r_1'$	$r_2+\Delta r_2$
$L_1+\Delta L$	$W_1+\Delta W$	$\lim_{\substack{\Delta r_1,\Delta r_1',\Delta r_2\to 0\\ \Delta p_1,\Delta p_1',\Delta p_2\to 0}} H_2(s)$	p_1	p_1	p_2	$\frac{r_1}{2}$	$\frac{r_1}{2}$	r_2

for different geometrical samples, varying from $N_{\min}$ to $N_{\max}$, we want to add new effective pole/residues without introducing non-uniqueness problem, such that ultimately the orders for all geometrical samples will be equal to the maximum effective order $N_{\max}$. The pole-residue tracking process is an iterative process. During this process, the minimum effective order is iteratively increased by one, i.e., after each iteration, the minimum effective order becomes $N_{\min} = N_{\min} + 1$. After a certain number of iterations, the minimum effective order becomes equal to the maximum effective order, i.e., $N_{\min} = N_{\max}$. The order of the transfer functions of all the geometrical samples then become the same as $N_{\max}$, i.e., $N_k = N_{\max}$, $\forall k \in T_r$.

More specifically, in each iteration, we search for all the samples to find those whose order is equal to $N_{\min}$. For each sample with order $N_{\min}$, say, the k^{th} sample, we choose one pole to be split into two overlapping poles according to the information of the neighboring sample. The neighboring sample (defined as the L^{th} sample) is the one which has the shortest geometrical distance to the k^{th} sample among all

the ones whose orders are equal to $N_{\min}+1$, defined as,

$$L = L(k) = \arg\min_{l \in T_s} \left\{ \sum_{i=1}^{n_x} \left\| \frac{x_i^{(l)} - x_i^{(k)}}{\delta_i} \right\| \right\} \tag{3.6}$$

where T_s is the sub-set containing all the training samples whose orders are equal to $N_{\min}+1$, and n_x is the number of the geometrical parameters. x_i is the i^{th} parameter in $\boldsymbol{x}$. δ_i is the smallest variance of x_i among all the samples, defined as,

$$\delta_i = \min_{\substack{l,k \in T_r \\ l \neq k}} \left\{ \left\| x_i^{(l)} - x_i^{(k)} \right\| \right\} \tag{3.7}$$

After (3.6), the L^{th} sample is selected as the neighbor of the k^{th} sample. A given sample may have more than one neighboring samples.

To determine which pole should be selected to be split, we perform $N_{\min}$ times of trials. In the m^{th} trial, we split the m^{th} pole and calculate the sum of differences between the imaginary parts of the present poles after splitting and the poles in the neighboring sample whose order is $N_{\min}+1$. Assume the order of the k^{th} geometrical sample is $N_k = N_{\min}$, while the L^{th} sample is the neighboring sample which has order $N_L = N_{\min}+1$. The formulation of the sum of differences is defined as follows

$$\begin{aligned} D_m = & \sum_{i=1}^{m} \left\| \mathrm{Im}(p_i^{(k)}) - \mathrm{Im}(p_i^{(L)}) \right\|^2 \\ & + \sum_{i=m}^{N_{\min}} \left\| \mathrm{Im}(p_i^{(k)}) - \mathrm{Im}(p_{i+1}^{(L)}) \right\|^2 \end{aligned} \tag{3.8}$$

This process is performed iteratively for $N_{\min}$ times with $m = 1, 2, ..., N_{\min}$. Because the poles should move continuously when the values of geometrical parameters change slightly, the correct splitting pole should be the one that leads to the mini-

mum sum of differences between the poles of the present sample (present geometrical sample) and the poles of the neighboring sample (whose values of geometrical parameters are slightly different from that of the present sample). Subsequently, we select the pole to be split according to the ranking of its D_m value. The pole with the minimum D_m value is selected as the splitting pole,

$$M = \arg \min_{m \in \{1, 2, ..., N_{\min}\}} \{D_m\} \tag{3.9}$$

The M^{th} pole of the k^{th} training sample is selected to be split since D_M is the minimum sum of differences among the various trials of pole-splitting. The splitting of pole p_M means that, as the geometrical parameters change, the pole p_M will be split into two new poles, whose values are both equal to p_M at the moment of splitting. Consequently, the M^{th} residue which is related to the selected pole should also be split into two separate residues. Each of the new residues has half the value of the original residue. The splitting of the M^{th} pole/residue is represented as follows

$$\frac{r_M}{s - p_M} = \frac{r'_M}{s - p_M} + \frac{r_{N_{\min}+1}}{s - p_{N_{\min}+1}} \tag{3.10}$$

where

$$\begin{aligned} p_{N_{\min}+1} &= p_M \\ r_{N_{\min}+1} &= r'_M = \frac{r_M}{2} \end{aligned} \tag{3.11}$$

Now the order of the transfer function of the k^{th} training sample has increased by one. Then we sort the poles and corresponding residues again to make the pole/residues in a continuous sequence. Using p'_j and r'_j to represent the poles and

residues after sorting, the sorting process is defined as follows

$$p'_j = \begin{cases} p_j, & j = 1, 2, ..., M \\ p_{N_{\min}+1}, & j = M+1 \\ p_{j-1}, & j = M+2, M+3, ..., N_{\min}+1 \end{cases}$$

$$r'_j = \begin{cases} r_j, & j = 1, 2, ..., M-1 \\ r'_j, & j = M \\ r_{N_{\min}+1}, & j = M+1 \\ r_{j-1}, & j = M+2, M+3, ..., N_{\min}+1 \end{cases} \tag{3.12}$$

The proposed pole-residue tracking technique resolves the discontinuity problem of pole-residues, illustrated in Table 3.3. Not only the values of $H_1(s)$ and $H_2(s)$ are the same, but also the pole/residues $\{P_1\ P_2\ P_3\ R_1\ R_2\ R_3\}$ in $H_1(s)$ and $H_2(s)$ approach each other, when ΔL and ΔW approach to zeros. As shown in Table 3.3, the pole/residues in each column of the table are continuous when the transfer function changes from $H_1(s)$ to $H_2(s)$ as $[L_1\ W_1]^T$ changes to $[L_1+\Delta L\ W_1+\Delta W]^T$.

This pole-residue tracking process is performed iteratively for all the samples whose order is equal to $N_{\min}$. After this process is finished, the minimum order $N_{\min}$ of the entire training set increases by one, i.e., $N_{\min} = N_{\min} + 1$. The pole-residue tracking process terminates when the minimum effective order is equal to the maximum effective order, i.e., $N_{\min} = N_{\max}$, thus, all the samples now have the same orders.

The overall pole-residue tracking algorithm is shown in the flowchart in Fig. 3.2. We implement this tracking technique by developing a C program based on the flowchart in Fig. 3.2 that automatically searches and splits the pole/residues for all the samples. The input file given to the program contains the minimum-order pole/residues produced from vector fitting for each geometrical sample. Such minimum-order may vary from sample to sample. The output file contains the constant-order pole/residues for all geometrical samples after the pole-residue tracking process.

Let vector $\boldsymbol{C}_k$ represent the poles and residues including both the original and newly added ones after the pole-residue tracking process

$$\begin{aligned} \boldsymbol{C}_k &= \begin{bmatrix} \boldsymbol{P}^{(k)} \\ \boldsymbol{R}^{(k)} \end{bmatrix} \\ &= [P_1^{(k)}\, P_2^{(k)} \cdots P_{N_{\max}}^{(k)}\, R_1^{(k)}\, R_2^{(k)} \cdots R_{N_{\max}}^{(k)}]^T \end{aligned} \tag{3.13}$$

where $\boldsymbol{P}^{(k)}$ is a vector of poles including both the original and newly added ones of the k^{th} sample, defined as $\boldsymbol{P}^{(k)} = [P_1^{(k)}\, P_2^{(k)} \cdots P_{N_{\max}}^{(k)}]^T$. $\boldsymbol{R}^{(k)}$ is a vector of residues including both the original and newly added ones of the k^{th} sample, defined as $\boldsymbol{R}^{(k)} = [R_1^{(k)}\, R_2^{(k)} \cdots R_{N_{\max}}^{(k)}]^T$. The pole-residue-based transfer function is rewritten as follows

$$H(\boldsymbol{C}_k, s) = \sum_{i=1}^{N_{\max}} \left(\frac{R_i^{(k)}}{s - P_i^{(k)}} + \frac{R_i^{*(k)}}{s - P_i^{*(k)}} \right) \tag{3.14}$$

where superscript "$*$" means complex conjugate. After order-changing process, we obtain the transfer functions of constant order. We use those poles and residues (including original and newly added ones) to build our proposed pole-residue-based

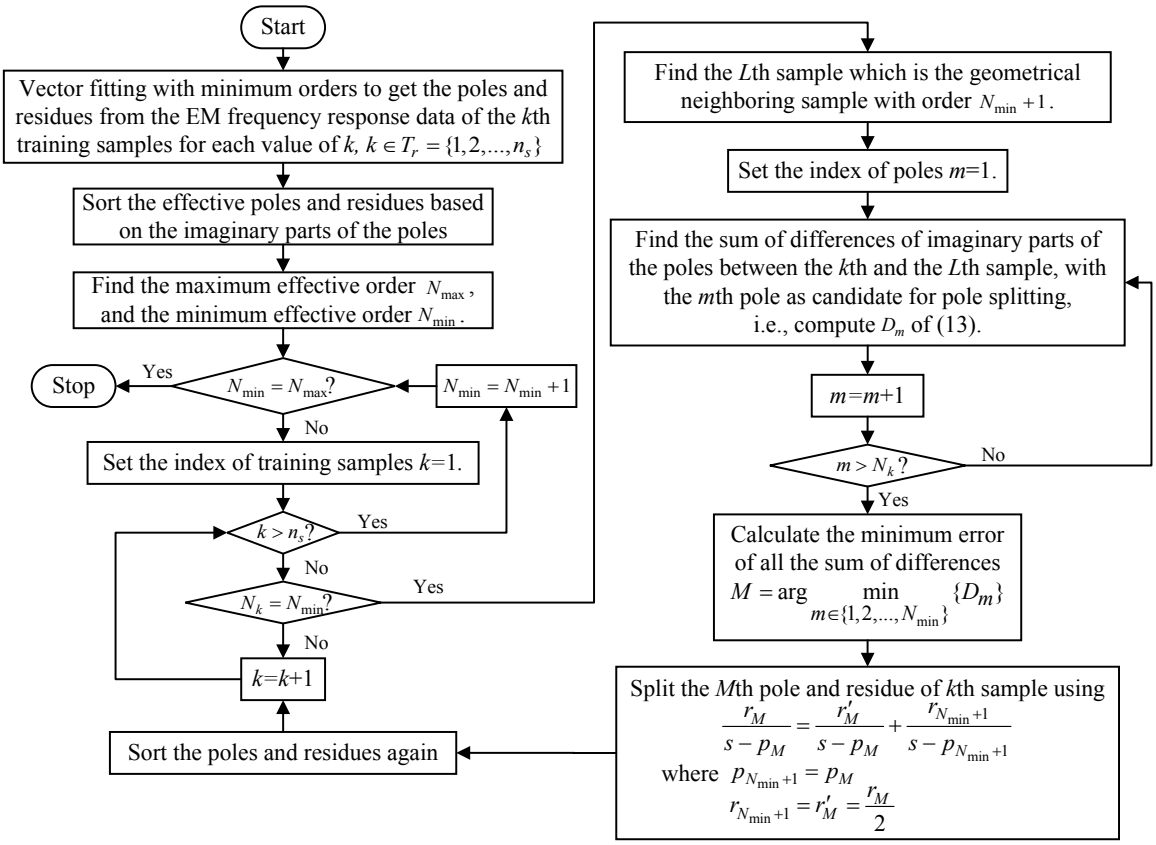


Figure 3.2: The flowchart of the overall pole-residue tracking algorithm to handle order-changing problem as geometrical parameters change.

neuro-TF model.

3.2.5 Preliminary Training of Neural Networks

We propose a two stage training process. In the first stage, we perform a preliminary training process. During this process, neural networks are trained to learn the relationships between all the pole/residues of the transfer function and the geometrical parameters. The training data for this phase is $(\boldsymbol{x}_k, \boldsymbol{C}_k)$, $k \in T_r$, i.e., samples of geometrical parameters as model inputs and pole/residues including both the

original and newly added ones as model outputs.

The nonlinear relationship between poles and geometrical parameters are usually different from that of residues. Subsequently, poles and residues are provided by separate neural networks. Let p_{NN} represent the outputs of the neural network for poles corresponding to geometrical variables $\boldsymbol{x}$ and neural network weights $\boldsymbol{w}_p$. Let r_{NN} represent the outputs of the neural network for residues corresponding to geometrical variables $\boldsymbol{x}$ and neural network weights $\boldsymbol{w}_r$. During the preliminary training, a certain amount of training error (e.g., 5%-10%) can be tolerated because the sensitivities of the transfer function with respect to the pole/residues are low and the solution can be further refined in next training phase. With this relaxed error criteria, less hidden neurons can be used in the neural networks. The neural networks thus have lower nonlinearity, which makes the pole-residue-based neuro-TF more robust. After preliminary training of the neural networks, an overall model refinement process is performed to further refine the final model described next.

3.2.6 Refinement Training of the Pole-Residue-Based Neuro-TF Model

In the second stage of the training process, a model refinement is performed to further refine the overall pole-residue-based neuro-TF model. The training data for this phase is $(\boldsymbol{x}_k,\ \boldsymbol{d}_k)$, $k \in T_r$, i.e., samples of geometrical parameters as model inputs and EM responses (i.e., S-parameters) as model outputs. The mechanism for the refinement training process of the overall pole-residue-based neuro-TF model is shown in Fig. 3.3. It consists of the pole-residue-based transfer function of (3.14)

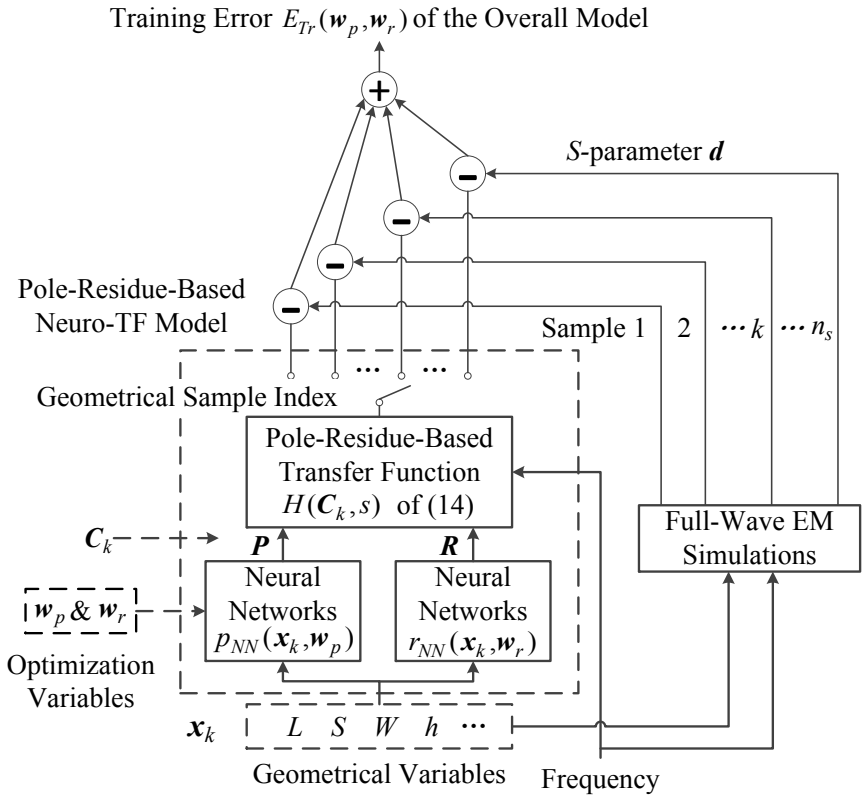


Figure 3.3: The mechanism for the refinement training process of the overall pole-residue-based neuro-TF model. The objective is to minimize the training error of the overall model. The variables of this training process are the weighting parameters in neural networks p_{NN} and r_{NN}.

and the neural networks whose initial values are the optimal solutions from the preliminary training. This model refinement process consists of both training and testing of the model. Training is performed by optimizing the weights inside the neural networks to minimize the error function

$$E_{Tr}(\boldsymbol{w}_p, \boldsymbol{w}_r) = \frac{1}{2n_s} \sum_{k \in T_r} \sum_{i \in \Omega} \| \boldsymbol{y}(p_{NN}(\boldsymbol{x}_k, \boldsymbol{w}_p), r_{NN}(\boldsymbol{x}_k, \boldsymbol{w}_r), s_i) - \boldsymbol{d}_{k,i} \|^2 \tag{3.15}$$

where T_r is the index set of training samples of various geometrical parameters, and n_s is the total number of training samples. Ω is the index set of frequency samples. $\boldsymbol{w}_p$ and $\boldsymbol{w}_r$ represent the weights in the neural networks for poles p_{NN} and residues r_{NN}, respectively. $\boldsymbol{y}$ represents the outputs of the overall model, which ultimately is a function of geometrical variables $\boldsymbol{x}_k$, frequency s_i, and neural network weights $\boldsymbol{w}_p$ and $\boldsymbol{w}_r$.

The training process terminates when the training error becomes lower than a user defined threshold E_t. After the training process, an independent set of testing data which are never used in training is used for testing the quality of the trained pole-residue-based neuro-TF model. The testing error E_{Te} is defined as the error between the model response and the testing data. If the testing error is also lower than the threshold error E_t, the model refinement process terminates and the pole-residue-based neuro-TF model is ready to be used for high-level design. Otherwise, the overall model training process will be repeated with different numbers of hidden neurons. A flowchart illustrating the overall pole-residue-based neuro-TF model development process is shown in Fig. 3.4.

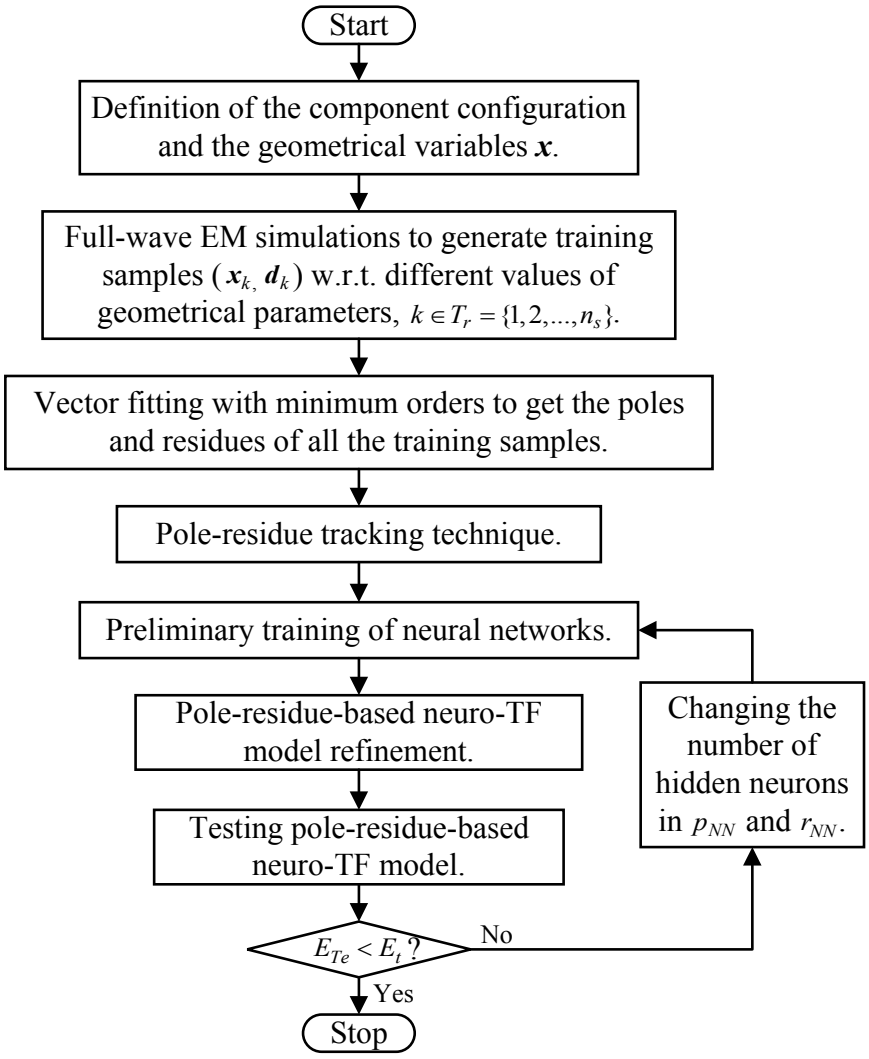


Figure 3.4: The flowchart of the overall pole-residue-based neuro-TF model development process.

3.3 Application Examples

3.3.1 Parametric Modeling of a Microwave Junction

In this example, we illustrate the development of pole-residue-based neuro-TF model for parametric modeling of the EM behavior of a microwave junction [14], as shown

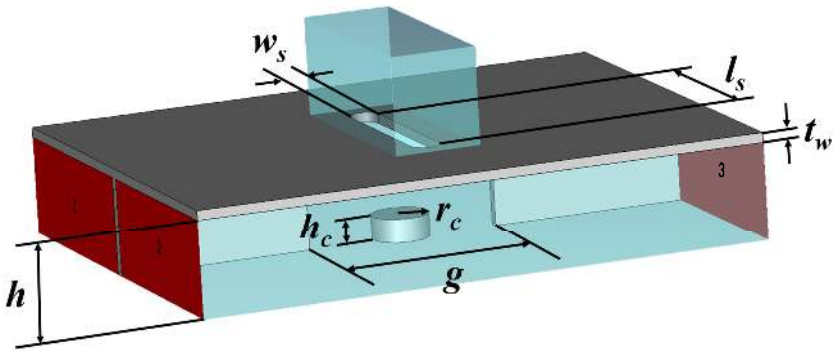


Figure 3.5: The geometrical parameters of the junction and the 3D configuration for EM simulation. A parametric model with respect to these seven geometrical parameters is to be developed.

in Fig. 3.5, where g is the gap between two conductive walls. h_c is the height of the tuning cylinder. r_c is the radius of the tuning cylinder. t_w is thickness of the conductive walls. w_s is width of the slot on the upper conductive wall. l_s is the length of the slot on the upper conductive wall. h is the height of the junction cavity.

This model has seven input geometrical variables, i.e., $\boldsymbol{x} = [g\ h_c\ r_c\ t_w\ w_s\ l_s\ h]^T$, as defined in the structure of the junction example, shown in Fig. 3.6. Frequency is an additional input. The model has eight outputs, i.e., $\boldsymbol{y} = [RS_{11}\ IS_{11}\ RS_{21}\ IS_{21}\ RS_{31}\ IS_{31}\ RS_{41}\ IS_{41}]^T$, which are the real and imaginary parts of S_{11}, S_{21}, S_{31} and S_{41}, respectively.

The *CST Studio Suite* 2014 software is used to perform the full-wave EM simulation and generate training and testing data for modeling. A parallel computation method is used in data generation. MPI is used for parallel processing with distributed memory. We use a cluster of *Dell PowerEdge* computers with *Intel Xeon*

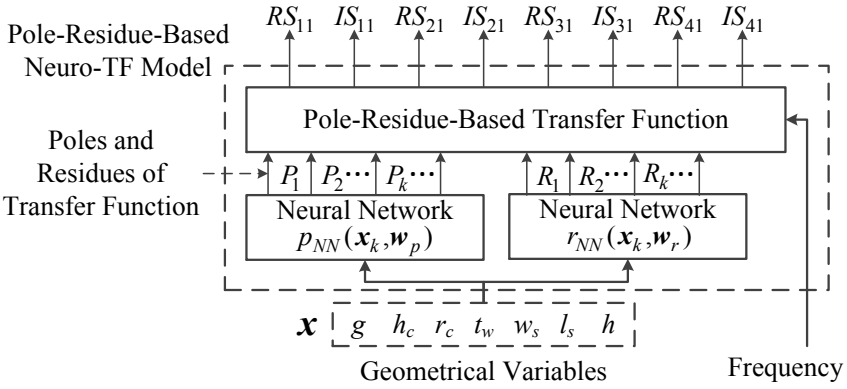


Figure 3.6: The structure of the pole-residue-based neuro-TF model for the junction example.

E5-2440 processors. We use six cores (two cores per computer from three computers) to generate the data. Design of experiments (DOE) [122] method is used as the sampling method for both training and testing data.

The proposed modeling method is applied to two different cases, i.e., Case 1 with a narrower parameter range and Case 2 with a wider parameter range as defined in Table 3.4. In Case 1, the minimum orders n_k of transfer functions stay unchanged at order eight for all samples, $k = 1, 2, ..., 81$, where the total number of geometrical samples is $n_s = 81$. In Case 2, the minimum orders n_k of transfer functions vary from six to eight among different samples of geometrical parameters, which results $N_{\min} = 3$ and $N_{\max} = 4$. In both cases, nine levels of DOE are used for defining samples of the training data, i.e., a total of 81 samples of training data, while seven levels of DOE are used for the testing data, i.e., a total of 49 samples of testing data. The ranges of training data and testing data for the two different cases are defined in Table 3.4. The total time for training data generation

Table 3.4: Definition of Training and Testing Data for The Junction Example

Geometrical Variables		Training Data (81 samples)			Testing Data (49 samples)		
		Min	Max	Step	Min	Max	Step
Case 1	g (mm)	19	19.8	0.1	19.05	19.75	0.1
	h_c (mm)	2	2.8	0.1	2.05	2.75	0.1
	r_c (mm)	3	3.8	0.1	3.05	3.75	0.1
	t_w (mm)	0.9	1.06	0.02	0.91	1.05	0.02
	w_s (mm)	3	3.8	0.1	3.05	3.75	0.1
	l_s (mm)	19.9	20.22	0.04	19.92	20.2	0.04
	h (mm)	10	10.8	0.1	10.05	10.75	0.1
Case 2	g (mm)	18	22	0.5	18.25	21.25	0.5
	h_c (mm)	2	3.6	0.2	2.1	3.3	0.2
	r_c (mm)	2.2	3.8	0.2	2.3	3.5	0.2
	t_w (mm)	0.84	1.16	0.04	0.86	1.1	0.04
	w_s (mm)	2.2	3.8	0.2	2.3	3.5	0.2
	l_s (mm)	19.6	20.4	0.1	19.65	20.25	0.1
	h (mm)	10	11.6	0.2	10.1	11.3	0.2

(for 81 samples of data) is about 15 minutes, while the total time for testing data generation (for 49 samples of data) is about nine minutes. If we use more computers to perform the parallel data generation, the time of data generation will be even shorter. The original frequency range is 7 GHz-9 GHz. We scale and shift the frequency range to 0.9 GHz-1.1 GHz. Pole-residue tracking technique for order-changing is applied. The pole-residue-based neuro-TF model is trained using the *NeuroModelerPlus* software. The average training error is 0.516%, while the average testing error is 0.558% for Case 2. The overall model training time is 9.6 minutes, including parameter extraction, preliminary training and refinement training.

For comparison purpose, we also apply brute-force pole-residue-based neuro-TF

modeling method which directly uses the maximum orders for all the samples of geometrical parameters and the recent bilinear rational neuro-TF modeling method with order-changing technique [28] for the two cases. Table 3.5 compares these various modeling methods in terms of ANN structures, coefficient (or pole/residue) continuity, coefficient (or pole/residue) sensitivity, and average training and testing error. In Case 1, since the geometrical parameters vary within a small range and the orders of transfer function stay unchanged, all methods obtain comparatively small training and testing errors. In Case 2, the geometrical parameters vary within a wider range and the orders of transfer function vary. The brute-force pole-residue-based neuro-TF modeling method has obvious discontinuity problems which result a high testing error. The overall model training time is 21.3 minutes for brute-force pole-residue-based neuro-TF modeling method using 15 hidden neurons. In the bilinear rational neuro-TF modeling method, the transfer function is more sensitive to its coefficients, thus the model needs more hidden neurons to achieve good training accuracy and cannot guarantee the testing error, where the values of geometrical parameters for testing are never used in training. The overall model training time is 18.6 minutes for bilinear rational neuro-TF modeling method using 15 hidden neurons.

When we use bilinear rational neuro-TF method, the transfer function is more sensitive to its coefficients, and the coefficients are more nonlinear than the pole/residues over geometrical parameters from sample to sample. When we use the brute-force method, the pole/residues have much discontinuity problem which also result in high nonlinearity of pole/residues from sample to sample. Thus the model trained

using the two methods cannot obtain a good training error with a small amount of hidden neurons. To achieve good training accuracy, we need to increase the number of hidden neurons in the model. In this example, however, such increase of the number of hidden neurons makes testing error even worse, further indicating the high-nonlinearity (or high sensitivity) of coefficient of transfer function with respect to changes in geometrical parameters, or high nonlinearity (or high sensitivity) of the pole/residues in brute-force method with respect to changes in geometrical parameters.

Our proposed method can achieve better accuracy in testing than the two other methods in the comparison. Fig. 3.7 shows the output S_{11} of the proposed pole-residue-based neuro-TF model for three different test geometrical samples of junctions, i.e., test geometrical samples #1, #2, and #3, and its comparison with model response using different methods and EM data. The geometrical variables for three samples of junctions are as follows.

Test geometrical sample #1: $\boldsymbol{x} = [18.25\ 2.1\ 2.3\ 0.86\ 2.3\ 19.65\ 10.1]^T$ (mm)

Test geometrical sample #2: $\boldsymbol{x} = [19.75\ 2.1\ 2.9\ 0.98\ 2.9\ 19.95\ 10.7]^T$ (mm)

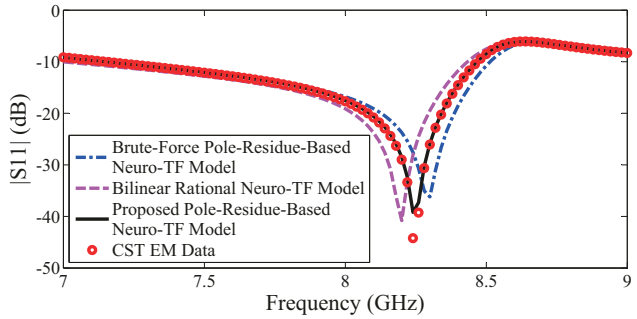
Test geometrical sample #3: $\boldsymbol{x} = [21.25\ 2.1\ 3.5\ 1.1\ 3.5\ 20.25\ 11.3]^T$ (mm)

It is observed that our model can achieve good accuracy for different geometrical samples even though these samples are never used in training. The comparisons of other outputs (i.e., S_{21}, S_{31}, and S_{41}) of the models developed using different modeling methods and EM data are similar with that of the output S_{11}. Once the pole-residue-based neuro-TF model training is completed, we can implement the trained model into the design optimization where the geometrical parameters can

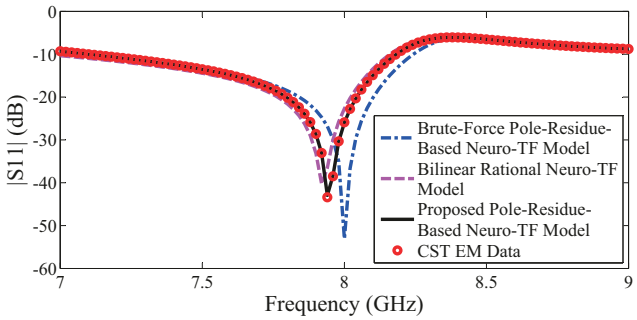
Table 3.5: Comparisons of Different Modeling Methods for The Junction Example

Training Method		Coefficient (or Pole-Residue) Continuity?	Coefficient (or Pole-Residue) Sensitivity	Number of Hidden Neurons	Average Training Error	Average Testing Error
Case 1 (Narrower Geometry Range)	Brute-Force Pole-Residue Neuro-TF Method	Yes	Low	4	0.258%	0.273%
	Bilinear Rational Neuro-TF Method	Yes	High	5	0.286%	0.292%
	Proposed Pole-Residue Neuro-TF Method	Yes	Low	4	0.254%	0.276%
Case 2 (Wider Geometry Range)	Brute-Force Pole-Residue Neuro-TF Method	No	Low	5	0.928%	1.568%
				15	0.528%	4.168%
	Bilinear Rational Neuro-TF Method	Yes	High	5	0.959%	1.237%
				15	0.559%	2.537%
	Proposed Pole-Residue Neuro-TF Method	Yes	Low	5	0.516%	0.558%

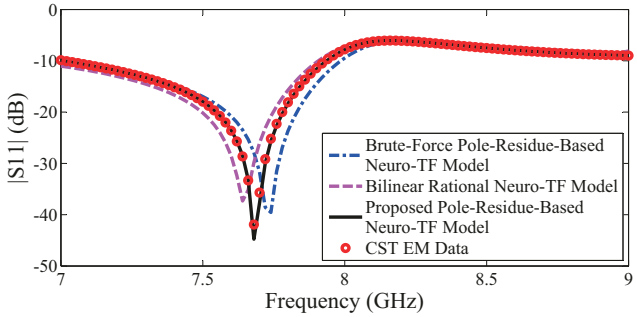
be repetitively adjusted during optimization. As an example of using the trained model for junction optimization, we perform EM optimization of three separate junctions with different design specifications:



(a)



(b)



(c)

Figure 3.7: Comparison of the magnitude in decibels of S_{11} of the models developed using different modeling methods and CST EM data: (a) test geometrical sample #1, (b) test geometrical sample #2, and (c) test geometrical sample #3 for the junction example. As shown in the figure, the proposed pole-residue-based neuro-TF model matches well with CST EM data even though the testing geometrical samples in the figures are never used in training.

Specifications for junction #1: $|S_{11}| \leq$ -40 dB, $|S_{21}| \leq$ -40 dB, $|S_{31}| \geq$ -3 dB, and $|S_{41}| \geq$ -3 dB at specified frequency of 7.8 GHz.

Specifications for junction #2: $|S_{11}| \leq$ -40 dB, $|S_{21}| \leq$ -40 dB, $|S_{31}| \geq$ -3 dB, and $|S_{41}| \geq$ -3 dB at specified frequency of 8 GHz.

Specifications for junction #3: $|S_{11}| \leq$ -40 dB, $|S_{21}| \leq$ -40 dB, $|S_{31}| \geq$ -3 dB, and $|S_{41}| \geq$ -3 dB at specified frequency of 8.1 GHz.

The initial values are $\boldsymbol{x} = [20\ 2.5\ 3\ 1\ 3\ 20\ 10]^T$ (mm). We perform design optimizations of the junction using the pole-residue-based neuro-TF model. The optimization took only about 20 seconds to achieve optimal solution for the each junction. The optimized geometrical values for the three separate junctions are:

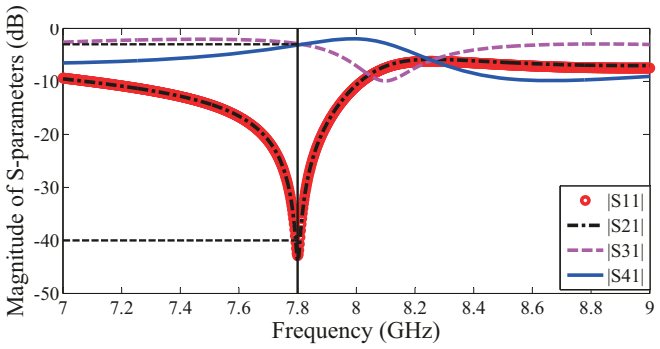
$\boldsymbol{x}_{opt}$ #1: $\boldsymbol{x} = [19.7995\ 3.17042\ 3.38173\ 0.950288\ 3.51267\ 20.2755\ 11.1084]^T$ (mm)

$\boldsymbol{x}_{opt}$ #2: $\boldsymbol{x} = [19.8\ 3.38746\ 3.10196\ 1.0438\ 2.58472\ 20.0805\ 10.6083]^T$ (mm)

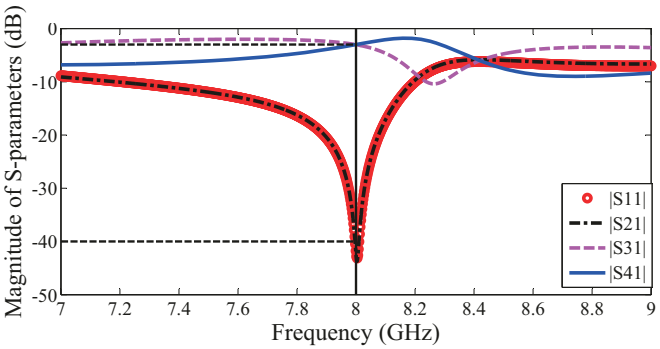
$\boldsymbol{x}_{opt}$ #3: $\boldsymbol{x} = [19.8\ 3.21313\ 3.21363\ 0.975674\ 2.43511\ 19.8437\ 10.5227]^T$ (mm)

The CST full-wave EM simulations at the model optimal solutions are shown in Fig. 3.8. For comparison purpose, we also use CST to perform direct EM optimization of the junction with the same initial value and same design specifications for the three junctions described above. The comparison is illustrated in Table 3.6.

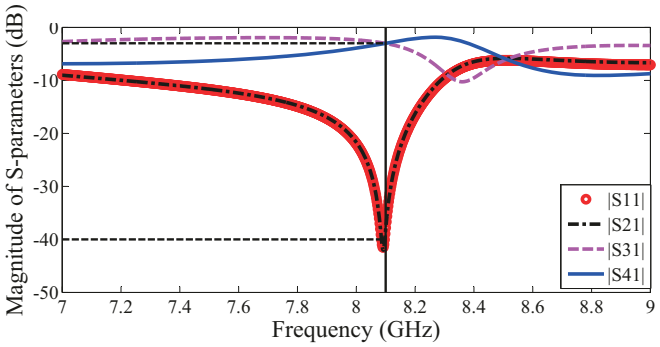
For pole-residue-based neuro-TF optimization, the model development time is 0.7 hours including EM data generation and model training. Once the model is trained, it can be re-used again and again for additional optimizations (i.e., for specifications for junctions 2 and 3). In Table 3.6, the total CPU time for model development and junction optimization using the pole-residue-based neuro-TF approach is 0.72 hours versus 7.7 hours for direct EM optimization. The more we



(a)



(b)



(c)

Figure 3.8: The magnitude in decibels of S_{11}, S_{21}, S_{31}, and S_{41} of CST EM data at (a) $\boldsymbol{x}_{opt}$ #1, (b) $\boldsymbol{x}_{opt}$ #2, and (c) $\boldsymbol{x}_{opt}$ #3, for three different sets of design specifications of the junction example. As shown in the figure, the proposed pole-residue-based neuro-TF model behaves well in design optimization with different specifications.

Table 3.6: CPU Time of Optimizations for The Junction Example

Different Specifications	CPU Time	
	Direct EM Optimization	Optimization Using Pole-Residue Neuro-TF Model
Specification 1	2.4 hours	0.7 hours (Model Development) + 20 s
Specification 2	2.2 hours	20 s
Specification 3	3.1 hours	20 s
Total	7.7 hours	0.72 hours

re-use the pole-residue-based neuro-TF model, the more time will be saved.

3.3.2 Parametric Modeling of an Ultra-Wideband Antenna

In this example, we illustrate the proposed technique using an ultra-wideband (UWB) antenna, as shown in Fig. 3.9, where R_l is the radius of the two large pads. d_l is the distance between the two large pads. t is the thickness of the substrate. g_s is the gap between the small pad and the large pad which are both connected to the port. r_s is the radius of the small pad.

The model has five input geometrical variables, i.e., $\boldsymbol{x} = [R_l\ d_l\ t\ g_s\ r_s]^T$, as defined in the structure of the UWB antenna example, shown in Fig. 3.10. Frequency is an additional input. The model has two outputs, i.e., $\boldsymbol{y} = [RS_{11}\ IS_{11}]^T$, which are the real and imaginary parts of S_{11}.

The *CST Studio Suite* 2014 software is used to perform the full-wave EM simulation. A parallel computation method with 6 cores is used in data generation. Design of experiments (DOE) method is used as sampling method.

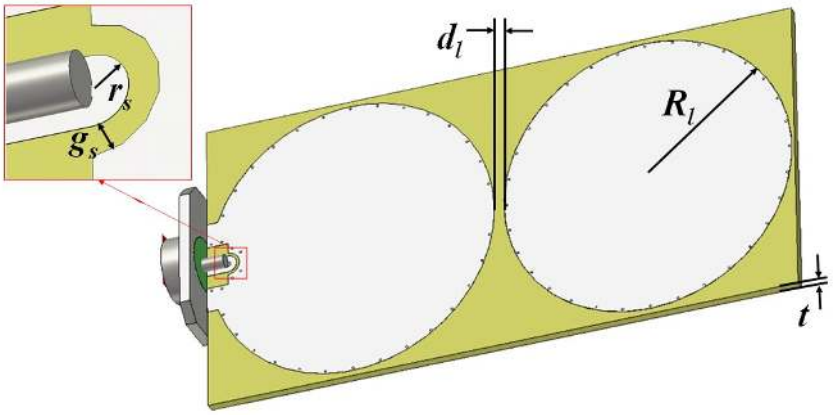


Figure 3.9: The geometrical parameters of the UWB antenna and the 3D configuration for EM simulation. A parametric model with respect to these five geometrical parameters is to be developed.

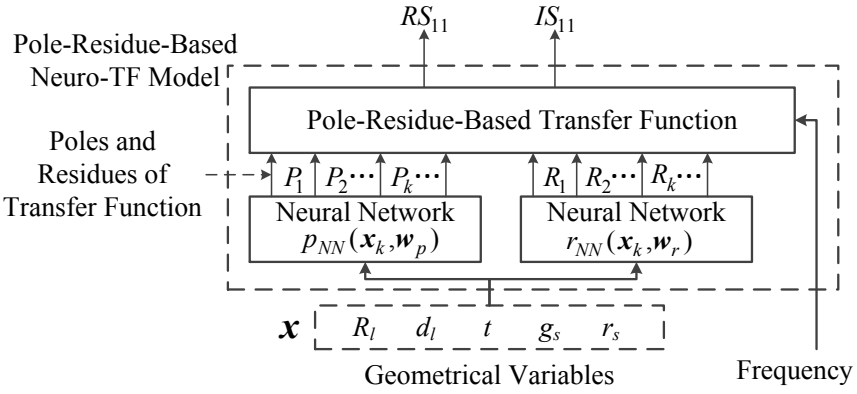


Figure 3.10: The structure of the pole-residue-based neuro-TF model for the UWB antenna example.

The proposed modeling method is applied to three different cases, i.e., Case 1 with a narrower parameter range, Case 2 and Case 3 with a wider parameter range, as defined in Table 3.7. The frequency range for Case 1 and Case 2 is 0-15 GHz, while the frequency range for Case 3 is 0-30 GHz. In Case 1, the minimum orders n_k of transfer functions vary slightly from 20 to 22, which results $N_{\min} = 10$

and $N_{\max} = 11$ among different samples of geometrical parameters, where the total number of geometrical samples is $n_s = 81$. In Case 2, the minimum orders n_k of transfer functions vary from 20 to 24, which results $N_{\min} = 10$ and $N_{\max} = 12$ among different samples of geometrical parameters. In Case 3, the minimum orders n_k of transfer functions vary from 32 to 36, which results $N_{\min} = 16$ and $N_{\max} = 18$ among different samples of geometrical parameters. In all the three cases, nine levels of DOE are used for defining samples of the training data, i.e., a total of 81 samples of training data, while seven levels of DOE are used for the testing data, i.e., a total of 49 samples of testing data. The ranges of training data and testing data are defined in Table 3.7. For Cases 1 and 2, the total time for training data generation (for 81 samples of data) is about 2.2 hours, while the total time for testing data generation (for 49 samples of data) is about 1.2 hours. The overall model training time for Case 2 is 23.4 minutes, including parameter extraction, preliminary training and refinement training. For Case 3, the total time for training and testing data generation is about 3.8 hours and 2.1 hours, respectively. For all the three cases, we scale and shift the frequency range to 0.9 GHz-1.1 GHz. The *NeuroModelerPlus* software is used for training the pole-residue-based neuro-TF model. The average training error for the three cases are shown in Table 3.7.

For comparison purpose, we also apply brute-force pole-residue-based neuro-TF modeling method which directly uses the maximum orders for all the samples of geometrical parameters and the recent bilinear rational neuro-TF modeling method with order-changing technique [28] for the three cases. Table 3.8 compares the model training and testing errors using these various modeling methods. In Case 1,

Table 3.7: Definition of Training and Testing Data for The UWB Antenna Example

Geometrical Variables		Training Data (81 samples)			Testing Data (49 samples)		
		Min	Max	Step	Min	Max	Step
Case 1	R_l (mm)	13.7	13.86	0.02	13.71	13.83	0.02
	d_l (mm)	0.9	1.06	0.02	0.91	1.03	0.02
	t (mm)	0.9	1.06	0.02	0.91	1.03	0.02
	g_s (mm)	0.4	0.56	0.02	0.41	0.53	0.02
	r_s (mm)	0.8	1.12	0.04	0.82	1.06	0.04
Case 2 & Case 3	R_l (mm)	13.4	14.2	0.1	13.45	14.05	0.1
	d_l (mm)	0.8	1.12	0.04	0.82	1.06	0.04
	t (mm)	0.8	1.12	0.04	0.82	1.06	0.04
	g_s (mm)	0.1	0.9	0.1	0.15	0.75	0.1
	r_s (mm)	0.2	1.8	0.2	0.3	1.5	0.2

all methods obtain comparatively small training and testing errors since the geometrical parameters vary within a small range. In Case 2, the geometrical parameters vary within a large range and the orders of transfer function have large variations. Both the brute-force pole-residue-based neuro-TF modeling method and the bilinear rational neuro-TF modeling method have high testing errors. The overall model training time is 54.1 minutes for brute-force pole-residue-based neuro-TF modeling method using 40 hidden neurons and 43.8 minutes for bilinear rational neuro-TF modeling method using 30 hidden neurons. Our proposed method can achieve better accuracy in testing than the two other methods in the comparison. Fig. 3.11 shows the outputs of the proposed pole-residue-based neuro-TF model for three different test geometrical samples of the UWB antenna, trained using data in Case 2, i.e., test geometrical samples #1, #2, and #3, and its comparison with model response

using different methods and EM data. The geometrical variables for three samples of UWB antennas are as follows.

Test geometrical sample #1: $\boldsymbol{x} = [13.55\ 0.82\ 0.86\ 0.25\ 0.50]^T$ (mm)

Test geometrical sample #2: $\boldsymbol{x} = [13.75\ 1.02\ 0.98\ 0.15\ 0.70]^T$ (mm)

Test geometrical sample #3: $\boldsymbol{x} = [14.05\ 0.90\ 0.86\ 0.15\ 1.30]^T$ (mm)

It is observed that our model can achieve good accuracy for different geometrical samples even though these samples are never used in training. In Case 3, we further increase the frequency range to 0-30 GHz, leading to high order transfer functions. Case 3 uses the same sets of training and testing geometrical samples as Case 2. Fig. 3.12 shows the outputs of the proposed model for the above three test geometrical samples, trained using data with wider frequency range in Case 3, and its comparison with model response using different methods and EM data. We can implement the trained model into the design optimization, after the pole-residue-based neuro-TF model training is completed. As an example of using the trained model for UWB antenna optimization, we perform design optimization of three separate UWB antennas with different design specifications:

Specifications for antenna #1: $|S_{11}| \leq$ -15 dB at frequency range of 2.7 GHz-10.6 GHz.

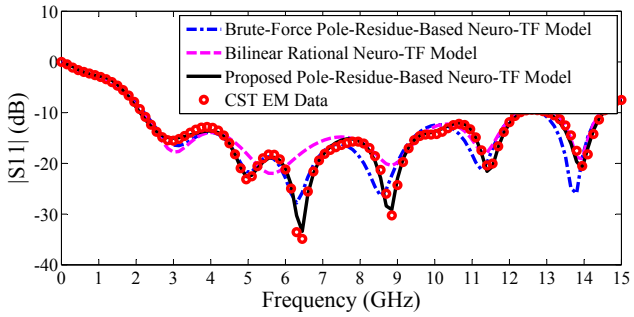
Specifications for antenna #2: $|S_{11}| \leq$ -15 dB at frequency range of 2.7 GHz-10.9 GHz.

Specifications for antenna #3: $|S_{11}| \leq$ -15 dB at frequency range of 2.7 GHz-11.3 GHz.

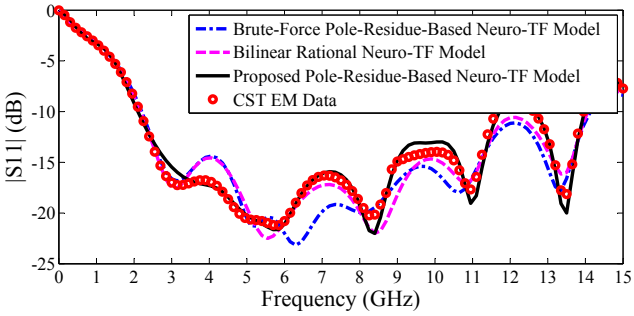
The initial values are $\boldsymbol{x} = [13.8\ 1\ 1\ 0.6\ 0.6]^T$ (mm). The design optimization

Table 3.8: Comparisons of Different Modeling Methods for The UWB Antenna Example

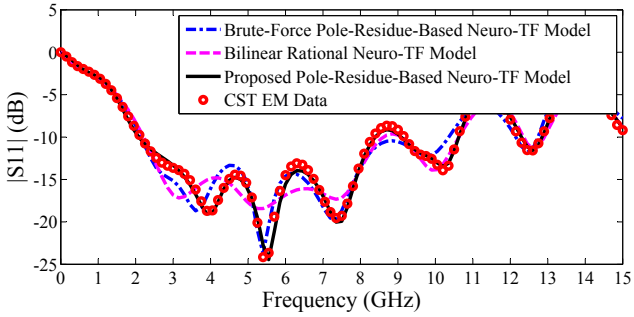
Training Method		Coefficient (or Pole-Residue) Continuity?	Coefficient (or Pole-Residue) Sensitivity	Number of Hidden Neurons	Average Training Error	Average Testing Error
Case 1 (Narrower Geometry Range)	Brute-Force Pole-Residue Neuro-TF Method	No	Low	5	0.415%	0.581%
	Bilinear Rational Neuro-TF Method	Yes	High	5	0.711%	0.771%
	Proposed Pole-Residue Neuro-TF Method	Yes	Low	5	0.319%	0.433%
Case 2 (Wider Geometry Range)	Brute-Force Pole-Residue Neuro-TF Method	No	Low	6	3.679%	10.19%
				40	1.4059%	35.06%
	Bilinear Rational Neuro-TF Method	Yes	High	6	3.719%	4.741%
				30	1.298%	16.38%
	Proposed Pole-Residue Neuro-TF Method	Yes	Low	6	1.225%	1.398%
Case 3 (Wider Frequency Range)	Brute-Force Pole-Residue Neuro-TF Method	No	Low	8	5.659%	13.215%
				50	1.792%	45.264%
	Bilinear Rational Neuro-TF Method	Yes	High	8	12.356%	25.326%
				60	1.864%	76.628%
	Proposed Pole-Residue Neuro-TF Method	Yes	Low	8	1.516%	1.872%



(a)

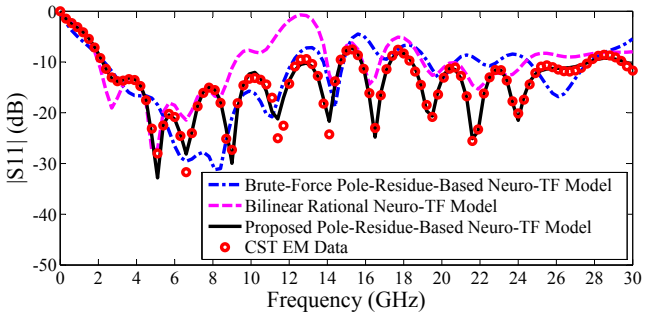


(b)

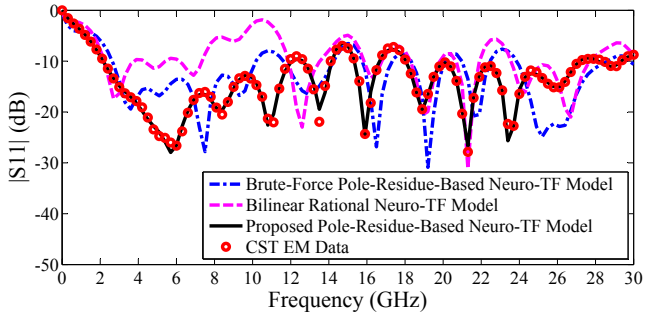


(c)

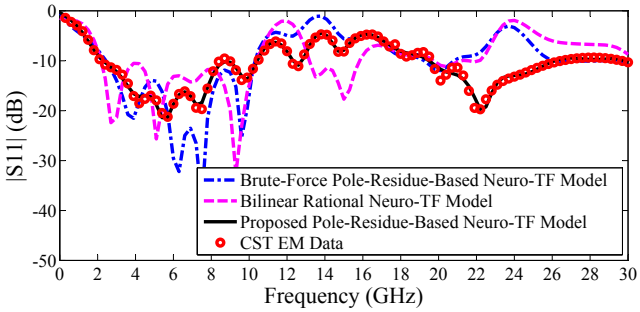
Figure 3.11: Comparison of the magnitude in decibels of S_{11} of the models developed using different modeling methods and CST EM data: (a) test geometrical sample #1, (b) test geometrical sample #2, and (c) test geometrical sample #3 for the UWB antenna example. As shown in the figure the proposed pole-residue-based neuro-TF model matches well with CST EM data even though the testing geometrical samples in the figures are never used in training.



(a)



(b)



(c)

Figure 3.12: Comparison of the magnitude of S_{11} of the models developed using different modeling methods and CST EM data in a wider frequency range of the UWB antenna example: (a) test geometrical sample #1, (b) test geometrical sample #2, and (c) test geometrical sample #3. As shown in the figure the proposed pole-residue-based neuro-TF model matches well with CST EM data even for responses with much higher order of the transfer function.

Table 3.9: CPU Time of Optimizations for The UWB Antenna Example

Different Specifications	CPU Time	
	Direct EM Optimization	Optimization Using Pole-Residue Neuro-TF Model
Specification 1	11.7 hours	3.9 hours (Model Development) + 20 s
Specification 2	10.3 hours	20 s
Specification 3	13.2 hours	20 s
Total	35.2 hours	3.92 hours

using the pole-residue-based neuro-TF model took only about 20 seconds to achieve the optimal design solution for the each UWB antenna. The optimized geometrical values for the three separate UWB antenna optimizations are:

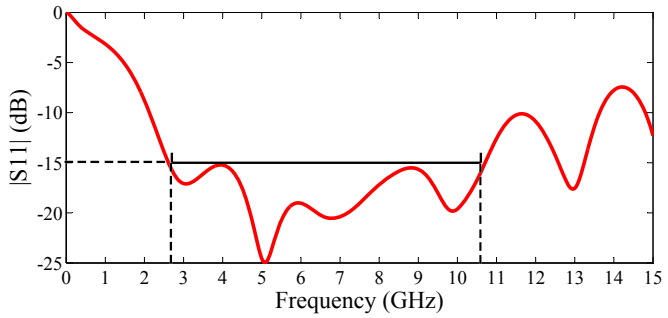
$\boldsymbol{x}_{opt}$ #1: $\boldsymbol{x} = [13.866\ 1.00512\ 0.938627\ 0.395753\ 0.796849]^T$ (mm)

$\boldsymbol{x}_{opt}$ #2: $\boldsymbol{x} = [13.802\ 0.986955\ 0.922442\ 0.289713\ 0.774063]^T$ (mm)

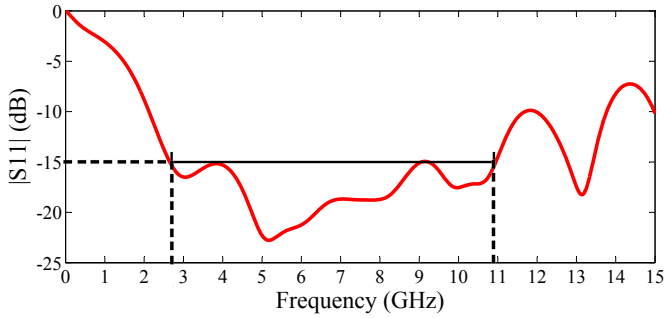
$\boldsymbol{x}_{opt}$ #3: $\boldsymbol{x} = [13.584\ 0.958071\ 0.913669\ 0.246313\ 0.699169]^T$ (mm)

The CST full-wave EM simulations at the model optimal solutions are shown in Fig. 3.13. For comparison purpose, we also use CST to perform direct EM optimization of the UWB antenna with the same initial value and same design specifications for the three UWB antennas described above. The comparison is illustrated in Table 3.9.

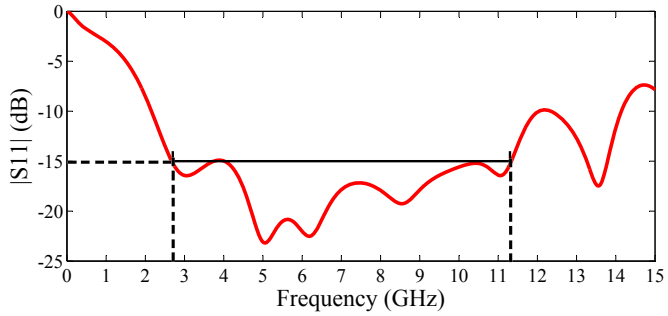
For pole-residue-based neuro-TF optimization, the model development time is 3.9 hours including EM data generation and model training. Once the model is trained, it can be re-used again and again for additional optimizations (i.e., for specifications for antennas 2 and 3). In Table 3.9, the total CPU time for model



(a)



(b)



(c)

Figure 3.13: The magnitude in decibels of S_{11} of CST EM data at (a) $\boldsymbol{x}_{opt}$ #1, (b) $\boldsymbol{x}_{opt}$ #2, and (c) $\boldsymbol{x}_{opt}$ #3, for three different sets of design specifications of the UWB antenna example. As shown in the figure, the proposed pole-residue-based neuro-TF model behaves well in design optimization with different specifications.

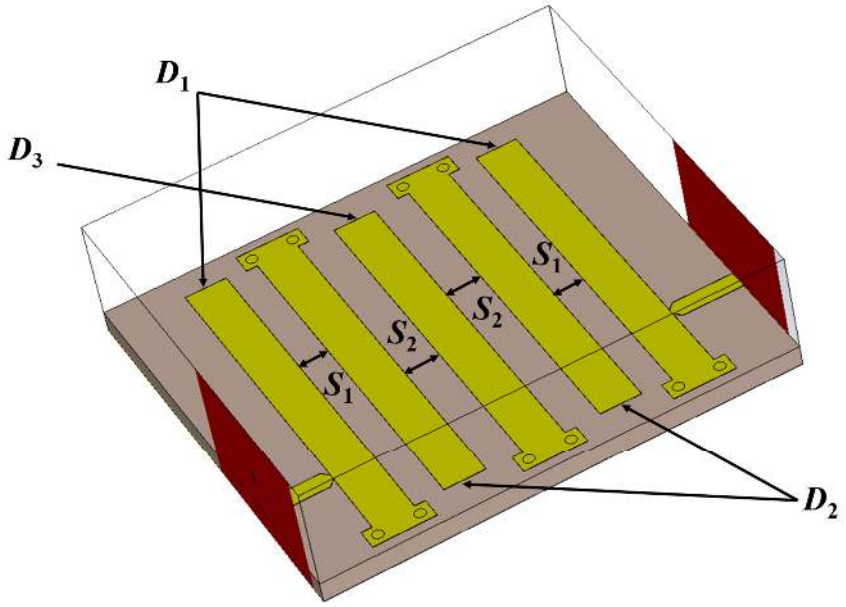


Figure 3.14: The geometrical parameters of the coupled-line filter and the 3D configuration for EM simulation. A parametric model with respect to these five geometrical parameters is to be developed.

development and UWB antenna optimization using the pole-residue-based neuro-TF approach is 3.92 hours versus 35.2 hours for direct EM optimization.

3.3.3 Parametric Modeling of a Coupled-Line Filter

In this example, we illustrate the proposed method for a coupled-line filter [14], as shown in Fig. 3.14, where S_1 and S_2 are the spacing between the microstrip lines. D_1, D_2, and D_3 are the offset distances from the ends of each lines to the corresponding fringes, respectively. The *CST Studio Suite* 2014 software is used to perform the full-wave EM simulation and generate training and testing data for modeling.

This model has five input geometrical variables, i.e., $\boldsymbol{x} = [S_1\ S_2\ D_1\ D_2\ D_3]^T$, as

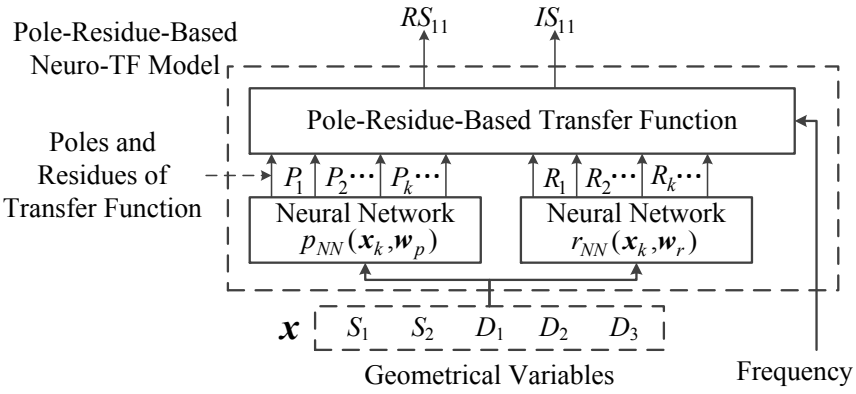


Figure 3.15: The structure of the pole-residue-based neuro-TF model for the coupled-line filter example.

defined in the structure of the filter example, shown in Fig. 3.15. Frequency is an additional input. The model has two outputs, i.e., $\boldsymbol{y} = [RS_{11}\ IS_{11}]^T$, which are the real and imaginary parts of S_{11}.

The proposed modeling method is applied to two different cases, i.e., Case 1 with a narrower parameter range and Case 2 with a wider parameter range as defined in Table 3.10. In Case 1, the minimum orders n_k of transfer functions vary slightly from eight to ten, which results $N_{\min} = 4$ and $N_{\max} = 5$ among different samples of geometrical parameters, where the total number of geometrical samples is $n_s = 81$. In Case 2, the minimum orders n_k of transfer functions vary from eight to twelve, which results $N_{\min} = 4$ and $N_{\max} = 6$ among different samples of geometrical parameters. In both cases, nine levels of DOE are used for defining samples of the training data, i.e., a total of 81 samples of training data, while seven levels of DOE are used for the testing data, i.e., a total of 49 samples of testing data. The ranges of training data and testing data are defined in Table 3.10. The total time for training

Table 3.10: Definition of Training and Testing Data for The Coupled-Line Filter Example

Geometrical Variables		Training Data (81 samples)			Testing Data (49 samples)		
		Min	Max	Step	Min	Max	Step
Case 1	S_1 (mm)	35	36.6	0.2	35.1	36.3	0.2
	S_2 (mm)	42	43.6	0.2	42.1	43.3	0.2
	D_1 (mm)	6	6.64	0.08	6.04	6.52	0.08
	D_2 (mm)	-2	-1.36	0.08	-1.96	-1.48	0.08
	D_3 (mm)	-2	-1.36	0.08	-1.96	-1.48	0.08
Case 2	S_1 (mm)	32	40	1	32.5	38.5	1
	S_2 (mm)	39	47	1	39.5	45.5	1
	D_1 (mm)	-4	12	2	-3	9	2
	D_2 (mm)	-12	4	2	-11	1	2
	D_3 (mm)	-12	4	2	-11	1	2

data generation (for 81 samples of data) is about one hour, while the total time for testing data generation (for 49 samples of data) is about 0.6 hours. The original frequency range is 2 GHz-3 GHz. We scale and shift the frequency range to 0.9 GHz-1.1 GHz. The overall model training takes 14.2 minutes for Case 2. The Q factor for this Coupled-Line Filter is around 420 for different geometrical samples, calculated using the EM simulator. After the modeling process, the average training error is 0.687%, while the average testing error is 0.954%.

For comparison purpose, we also apply brute-force pole-residue-based neuro-TF modeling method which directly uses the maximum orders for all the samples of geometrical parameters and the recent bilinear rational neuro-TF modeling method with order-changing technique [28] for the two cases. Table 3.11 compares these various modeling methods. In Case 1, since the geometrical parameters vary within

a small range and the orders of transfer function change slightly, all methods obtain comparatively small training and testing errors. In Case 2, when the geometrical parameters vary within a wider range and the orders of transfer function have larger variations, The brute-force pole-residue-based neuro-TF modeling method and the bilinear rational neuro-TF modeling method has a high testing error. The overall model training takes 32.6 minutes and 28.3 minutes for the brute-force pole-residue-based method and the bilinear rational method, respectively. Our proposed method can achieve better accuracy in testing than the two other methods in the comparison. Fig. 3.16 shows the outputs of the proposed pole-residue-based neuro-TF model for three different test geometrical samples of filters, i.e., test geometrical samples #1, #2, and #3, and its comparison with model response using different methods and EM data. The geometrical variables for three samples of filters are as follows.

Test geometrical sample #1: $\boldsymbol{x} = [32.5\ 45.5\ 9\ -9\ -7]^T$ (mm)

Test geometrical sample #2: $\boldsymbol{x} = [35.5\ 39.5\ 3\ -5\ -5]^T$ (mm)

Test geometrical sample #3: $\boldsymbol{x} = [38.5\ 45.5\ 1\ -3\ -9]^T$ (mm)

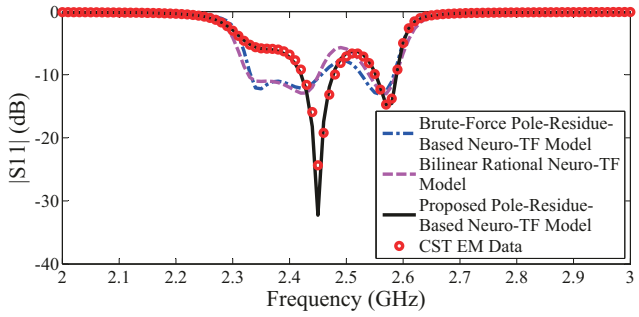
It is observed that our model can achieve good accuracy for different geometrical samples even though these samples are never used in training.

As another case to demonstrate the proposed technique, we further change the metal of the microstrip lines into perfect electric conductor (PEC), increasing the Q factor for this Coupled-Line Filter to around 580 for different geometrical samples. The same sets of geometrical samples as in Case 2 in Table 3.10 is used for generating training and testing data for training and validating the model of this higher Q filter. The slopes of S-parameters around pole frequencies for this higher Q filter

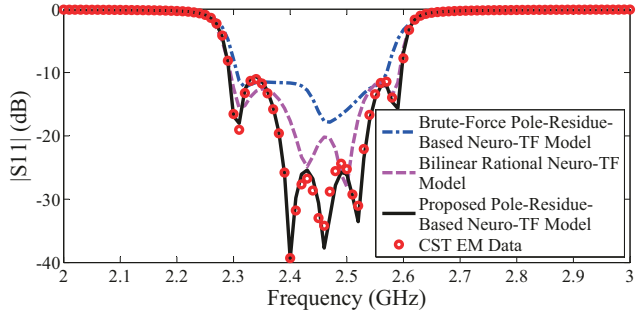
Table 3.11: Comparisons of Different Modeling Methods for The Coupled-Line Filter Example

	Training Method	Coefficient (or Pole-Residue) Continuity?	Coefficient (or Pole-Residue) Sensitivity	Number of Hidden Neurons	Average Training Error	Average Testing Error
Case 1 (Narrower Geometry Range)	Brute-Force Pole-Residue Neuro-TF Method	No	Low	5	0.624%	0.778%
	Bilinear Rational Neuro-TF Method	Yes	High	5	0.685%	0.856%
	Proposed Pole-Residue Neuro-TF Method	Yes	Low	5	0.598%	0.686%
Case 2 (Wider Geometry Range)	Brute-Force Pole-Residue Neuro-TF Method	No	Low	5	6.556%	10.23%
				30	0.732%	27.91%
	Bilinear Rational Neuro-TF Method	Yes	High	5	5.906%	7.245%
				30	0.747%	10.02%
	Proposed Pole-Residue Neuro-TF Method	Yes	Low	5	0.687%	0.954%

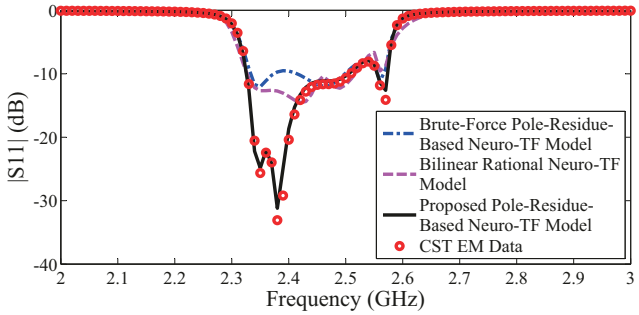
are steeper than those of the original filter. After the modeling process, the average training error is 0.725%, while the average testing error is 0.917%. Fig. 3.17 shows the outputs of the proposed pole-residue-based neuro-TF model of the higher Q



(a)



(b)



(c)

Figure 3.16: Comparison of the magnitude in decibels of S_{11} of the models developed using different modeling methods and CST EM data: (a) test geometrical sample #1, (b) test geometrical sample #2, and (c) test geometrical sample #3 for the coupled-line filter example. As shown in the figure the proposed pole-residue-based neuro-TF model matches well with CST EM data even though the testing geometrical samples used in the figures are never used in training.

filter and original filter responses for the three test geometrical samples of filters defined above, and their comparisons with EM data.

It is observed that similar training and testing error are achieved for both higher Q filter and original filter. By utilizing pole-residue-based transfer functions as part of the neuro-TF model, this technique can model examples for low Q or high Q response.

Once the pole-residue-based neuro-TF model training is completed, we can implement the trained model into the design optimization where the geometrical parameters can be repetitively adjusted during optimization. As an example of using the trained model for filter optimization, we perform optimization of three separate filters with different design specifications:

Specifications for filter #1: $|S_{11}| \leq$ -20 dB at frequency range of 2.3 GHz-2.54 GHz.

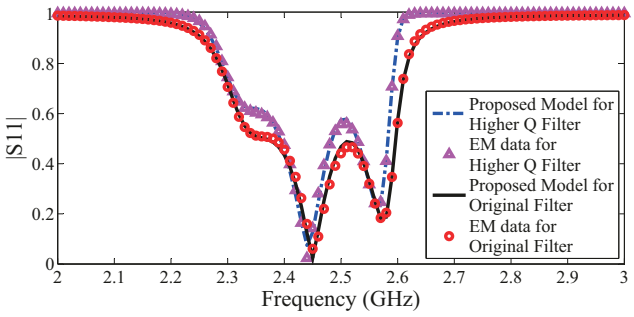
Specifications for filter #2: $|S_{11}| \leq$ -20 dB at frequency range of 2.32 GHz-2.56 GHz.

Specifications for filter #3: $|S_{11}| \leq$ -20 dB at frequency range of 2.34 GHz-2.58 GHz.

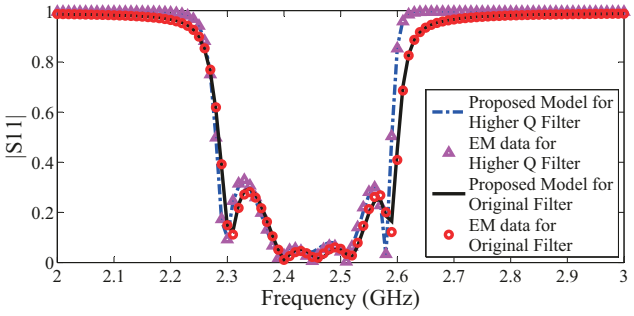
The initial values are $\boldsymbol{x} = [36\ 43\ 6\ -2\ -2]^T$ (mm). We perform design optimizations of the coupled-line filter using the pole-residue-based neuro-TF model. The optimization took only about 20 seconds to achieve optimal solution for the each filter. The optimized geometrical values for the three separate filter solutions are:

$\boldsymbol{x}_{opt}$ #1: $\boldsymbol{x} = [36.8889\ 44.068\ 8.10735\ -0.120832\ 0.236612]^T$ (mm)

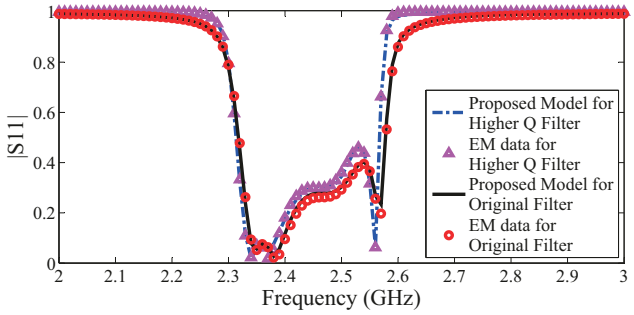
$\boldsymbol{x}_{opt}$ #2: $\boldsymbol{x} = [36.4979\ 43.7801\ 4.61309\ -3.9472\ -3.62018]^T$ (mm)



(a)



(b)



(c)

Figure 3.17: Comparison of the magnitude of S_{11} of the proposed model of the higher Q filter and original filter responses and EM data for the coupled-line filter example: (a) test geometrical sample #1, (b) test geometrical sample #2, and (c) test geometrical sample #3. As shown in the figure the proposed pole-residue-based neuro-TF model matches the EM data of both higher Q and original filter response well.

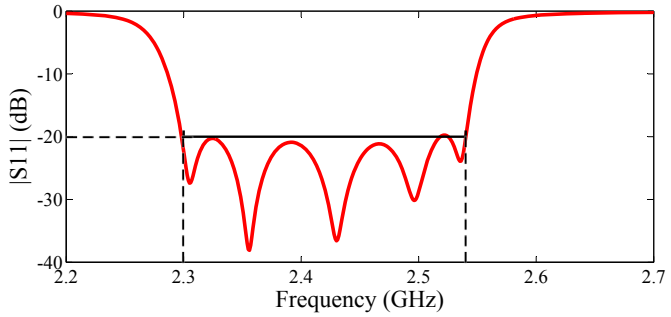
Table 3.12: CPU Time of Optimizations for The Coupled-Line Filter Example

Different Specifications	CPU Time	
	Direct EM Optimization	Optimization Using Pole-Residue Neuro-TF Model
Specification 1	6.3 hours	1.93 hours (Model Development) + 20 s
Specification 2	7.8 hours	20 s
Specification 3	8.6 hours	20 s
Total	22.7 hours	1.95 hours

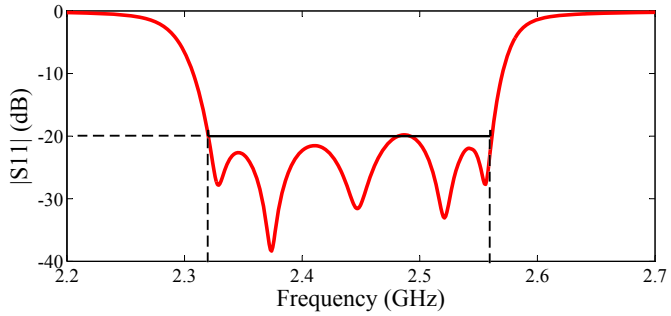
$\boldsymbol{x}_{opt}$ #3: $\boldsymbol{x}$ = [36.0017 43.5694 0.948699 -7.83401 -7.35138]^T (mm)

The CST full-wave EM simulations at the model optimal solutions are shown in Fig. 3.18. For comparison purpose, we also use CST to perform direct EM optimization of the filter with the same initial value and same design specifications for the three filters described above. The comparison is illustrated in Table 3.12.

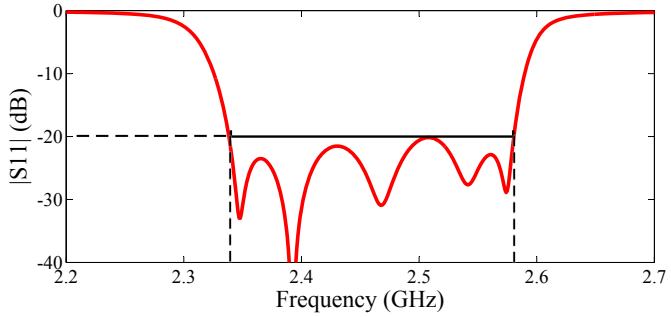
For pole-residue-based neuro-TF optimization, the model development time is 1.93 hours including EM data generation and model training. Once the model is trained, it can be re-used again and again for additional optimizations (i.e., for specifications for filters 2 and 3). In Table 3.12, the total CPU time for model development and filter optimization using the pole-residue-based neuro-TF approach is 1.95 hours versus 22.7 hours for direct EM optimization. The more we re-use the pole-residue-based neuro-TF model, the more time will be saved.



(a)



(b)



(c)

Figure 3.18: The magnitude in decibels of S_{11} of CST EM data at (a) $\boldsymbol{x}_{opt}$ #1, (b) $\boldsymbol{x}_{opt}$ #2, and (c) $\boldsymbol{x}_{opt}$ #3, for three different sets of design specifications of the coupled-line filter example. As shown in the figure, the proposed pole-residue-based neuro-TF model behaves well in design optimization with different specifications.

3.4 Conclusion

In this chapter, a new approach to parametric modeling of EM behavior of microwave components using combined neural networks and pole-residue-based transfer function has been proposed. In the proposed method, a novel pole-residue tracking technique has been developed for pole-residue-based transfer function to deal with the problem of changing orders when geometrical parameters undergo large variations. With high dimension of geometrical parameter space and large geometrical variations, the model obtained by the proposed method is more accurate and robust than that obtained by the existing modeling method using rational transfer function formulation. The proposed method can also obtain good modeling accuracy even when the orders of the models are high. Once trained, the pole-residue-based neuro-TF model developed by the proposed method provides fast and accurate prediction of the EM responses w.r.t the geometrical variables. The developed pole-residue-based neuro-TF models can be also used for the high-level optimizations with geometrical parameters as design variables.

Chapter 4

Advanced Pole-Residue Tracking Technique Using EM Sensitivity Analysis

This chapter proposes an advanced pole-residue tracking technique using EM sensitivity analysis. The purpose is to exploit the sensitivity information to track the splitting of poles as geometrical parameters change. This pole-residue tracking technique allows the model to bridge the differences of the orders of transfer function over different regions of the geometrical parameters, and ultimately form smooth and continuous functions between the pole/residues and the geometrical variables. The proposed technique addresses the challenges of tracking pole splitting when training data are limited.

4.1 Introduction

The discontinuity of pole/residues in transfer functions with respect to the geometrical variables is a major issue for modeling. When geometrical variations are large,

the corresponding EM responses will lead to different orders of transfer functions. A repetitive parameter extraction process is required to extract the pole-residues of transfer functions for each sample of geometrical variables. One way is to set the orders to the maximum one among all geometrical samples. In that case, high order transfer function is used to deal with low order problems for subsets of geometrical samples. That causes non-unique and arbitrary numerical solutions for the vector fitting process, resulting in the discontinuity of pole/residues.

Another way is to use a minimum order of the pole-residue-based transfer function for each geometrical sample [29]. That may result in different orders of transfer functions for different geometrical samples. These varied orders will result in abrupt changes of the values of poles and residues, and/or abrupt changes in the function structure of poles and residues as functions of geometrical parameters. These abrupt changes also lead to the discontinuity of pole/residues.

In the previous chapter, we introduce a pole-residue tracking technique to overcome the discontinuity problems of pole/residues in transfer functions. The idea is based on splitting one pole into two separate new poles to achieve the increase of the order of the pole-residue-based transfer function. This approach generates a group of new pole/residues based on those in the minimum order transfer functions and ultimately form transfer functions of constant order. The method solves the discontinuity problems of pole/residues while maintaining the accuracy of the transfer functions.

This chapter presents a further advance over the work of the pole-residue tracking technique. An advanced pole-residue tracking technique using EM sensitivity

analysis is proposed in this chapter. The purpose is to exploit sensitivity information to solve the challenge of pole splitting especially when the amount of training data are reduced and/or the geometrical step sizes between the data samples are enlarged. This proposed technique takes advantage of sensitivity information to split one pole into two separate new poles to achieve the increase of the order of transfer function and ultimately form transfer functions of constant order over the entire region of geometrical parameters. The proposed technique addresses the challenges of tracking pole splitting when training data are limited.

4.2 Advanced Pole-Residue Tracking Technique Using EM Sensitivity Analysis

The transfer functions in pole/residue format are used in the proposed neuro-TF model. Poles and residues of the transfer function will change as the geometrical parameters change. Because the relationship between the pole/residues and the geometrical parameters is non-linear and unknown, neural networks are used to represent the relationship. Let $\boldsymbol{x}$ represent the input vector of the neuro-TF model such as geometrical parameters of microwave passive components. Let $\boldsymbol{y}$ represent the outputs of the neuro-TF model such as S-parameters. The outputs $\boldsymbol{y}$ contains the real/imaginary parts of transfer function response H, which is formulated in pole/residue format [30],

$$H(s, \boldsymbol{x}, \boldsymbol{w}) = \sum_{i=1}^{N} \frac{r_i(\boldsymbol{x}, \boldsymbol{w})}{s - p_i(\boldsymbol{x}, \boldsymbol{w})} + \sum_{i=1}^{N} \frac{r_i^*(\boldsymbol{x}, \boldsymbol{w})}{s - p_i^*(\boldsymbol{x}, \boldsymbol{w})} \tag{4.1}$$

where N represents the effective order of transfer function. p_i and r_i represent the effective poles and residues, respectively. The superscript "$*$" means complex conjugate. s represents the frequency in Laplace domain. $\boldsymbol{w}$ represents the weighting parameters in the neural networks. Let $\boldsymbol{p}$ and $\boldsymbol{r}$ represent vectors containing all the poles and residues, respectively. Separate neural networks are used to represent the real and imaginary parts of the pole/residues in order to maintain real value neural network structures, defined as

$$\begin{aligned} \boldsymbol{p}(\boldsymbol{x}, \boldsymbol{w}) &= \boldsymbol{\alpha}(\boldsymbol{x}, \boldsymbol{w}) + j\boldsymbol{\beta}(\boldsymbol{x}, \boldsymbol{w}) \\ \boldsymbol{r}(\boldsymbol{x}, \boldsymbol{w}) &= \boldsymbol{\eta}(\boldsymbol{x}, \boldsymbol{w}) + j\boldsymbol{\xi}(\boldsymbol{x}, \boldsymbol{w}) \end{aligned} \tag{4.2}$$

where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ represent the vectors containing the real and imaginary parts of the poles $\boldsymbol{p}$, respectively, while $\boldsymbol{\eta}$ and $\boldsymbol{\xi}$ represent the vectors containing the real and imaginary parts of the residues $\boldsymbol{r}$, respectively. $\boldsymbol{w}$ represents the weighting parameters in the neural networks. $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\eta}$, and $\boldsymbol{\xi}$ represent the functions of neural networks with respect to $\boldsymbol{x}$ and $\boldsymbol{w}$.

4.2.1 Pole-Residue Extraction

In the proposed method, pole-residue-based transfer function formulation is used for the proposed neuro-TF model. Firstly, we perform parameter extraction to obtain the data for preliminary training of the neural networks $\boldsymbol{\alpha}(\boldsymbol{x}, \boldsymbol{w})$, $\boldsymbol{\beta}(\boldsymbol{x}, \boldsymbol{w})$, $\boldsymbol{\eta}(\boldsymbol{x}, \boldsymbol{w})$, and $\boldsymbol{\xi}(\boldsymbol{x}, \boldsymbol{w})$. Pole-residue extraction is to obtain the data of poles and residues with respect to geometrical parameters $\boldsymbol{x}$ for training the original neural networks.

Different samples of geometrical parameters, which are also called as training

samples, are generated with respect to the inputs $\boldsymbol{x}$ containing all the geometrical parameters except frequency which is a separate variable swept by the EM simulator during data generation. Let $\boldsymbol{d}$ represent the data of the outputs generated from EM simulations (e.g., S-parameters). Let $\boldsymbol{d}'$ represent the derivative data of the outputs with respect to geometrical parameters from EM sensitivity analysis. EM derivative data $\boldsymbol{d}'$ is generated using EM sensitivity analysis simultaneously with the generation of EM output data $\boldsymbol{d}$ in the EM simulator. Let T_r represent the index set of the training samples of different values of geometrical parameters, i.e., $T_r = \{1, 2, ..., n_s\}$, where n_s is the total number of the training samples of geometrical parameters.

Pole-residue extraction is performed using a vector fitting process [121]. We use vector fitting to obtain a group of effective poles and residues from the EM output data $\boldsymbol{d}$ versus frequency, using the minimum order of transfer function for each geometrical sample. With minimum order of transfer function, we can avoid non-unique and arbitrary numerical solutions of pole/residues from the vector fitting process. However, by doing so, the orders of transfer function may vary among different geometrical samples. Let N_k represent the effective order of transfer function for the k^{th} geometrical sample, where k represents the index of the geometrical samples, i.e., $k \in T_r$. N_k is initially equal to the minimum effective order of transfer function for the k^{th} sample determined from the vector fitting process.

To make the output data for the neural network real values, the real and imaginary parts of the effective poles and residues are used in our formulation to form the transfer function. Let $\tilde{\boldsymbol{\alpha}}$ and $\tilde{\boldsymbol{\beta}}$ represent the real and imaginary parts of the poles

of the transfer function whose order may vary among different geometrical samples. Let $\tilde{\boldsymbol{\eta}}$ and $\tilde{\boldsymbol{\xi}}$ represent the real and imaginary parts of the corresponding residues, respectively. Let $\tilde{\boldsymbol{c}}$ represent a vector containing these real and imaginary parts of poles and residues, defined as

$$\tilde{\boldsymbol{c}} = \left[\tilde{\boldsymbol{\alpha}}^T \ \tilde{\boldsymbol{\beta}}^T \ \tilde{\boldsymbol{\eta}}^T \ \tilde{\boldsymbol{\xi}}^T\right]^T \tag{4.3}$$

Let $\tilde{\boldsymbol{c}}_k$ represent $\tilde{\boldsymbol{c}}$ at a particular sample, i.e., the k^{th} sample, containing $\tilde{\boldsymbol{\alpha}}^k$, $\tilde{\boldsymbol{\beta}}^k$, $\tilde{\boldsymbol{\eta}}^k$ and $\tilde{\boldsymbol{\xi}}^k$, which represent real and imaginary parts of poles and residues at the k^{th} sample.

During the calculation of $\tilde{\boldsymbol{c}}_k$, we use minimum orders of the transfer function for each geometrical sample to avoid non-unique and arbitrary numerical solutions of pole/residues from the vector fitting process. However, that may result in varied orders of the transfer function for different geometrical samples. The varied orders can lead to abrupt changes in values of poles and residues, and/or abrupt changes in the function structure of poles and residues as geometrical parameters change. These abrupt changes will lead to the discontinuity of pole/residues with respect to changes in geometrical parameters. The technique dealing order changing problem for the rational format of neuro-transfer function in the references [28], [32] is not applicable for the pole-residue based neuro-transfer function used in this chapter. An advanced pole-residue tracking technique is proposed in the next sub-section to overcome the discontinuity problem of pole/residues.

4.2.2 Advanced Pole-Residue Tracking Technique Using Derivative Information

The orders of transfer function may vary over different values of geometrical parameters. For example, the number of poles may increase when the geometrical parameters are changed. In order to track how the few poles change to more poles when the geometrical parameters change, we derive a pole splitting method where one of the poles is split into two new poles as geometrical parameters are changed. In this sub-section, an advanced pole-residue tracking technique using derivative information is proposed to perform the pole splitting.

After pole-residue extraction and derivative data calculation, we obtain $\tilde{\boldsymbol{c}}_k$ and $\boldsymbol{A}_k$ for each geometrical sample, i.e., the k^{th} sample. By using pole splitting, we want to increase the effective orders of transfer function N_k by adding new effective pole/residues without introducing non-uniqueness problem, such that the orders of transfer function for all geometrical samples will ultimately be equal to the maximum effective order N. By doing so, the pole/residues become continuous with respect to changes in geometrical parameters. The maximum effective order N is defined as

$$N = \max_{k \in T_r} \{N_k\} \tag{4.4}$$

The pole-residue tracking process is an iterative process. In the first iteration, the effective order of the transfer function N_k is initially set to be equal to the minimum effective order from the vector fitting solutions for the k^{th} sample. During the pole-residue tracking process, we iteratively select the geometrical samples which have the lowest order of transfer function among all the geometrical samples. The

effective order of transfer function of those samples is increased by one, while the pole/residues are formulated to be continuous with respect to changes in geometrical parameters. After the iterative process, the orders of transfer function for all geometrical samples are equal to the maximum effective order N. In each iteration of the iterative process, let $\mathbf{\Gamma}$ contain the indices of the geometrical samples which have the lowest order of transfer function among all the geometrical samples, defined as,

$$\mathbf{\Gamma} = \left\{ k \,\middle|\, k = \arg\min_{l \in T_r} \{N_l\} \right\} \tag{4.5}$$

Among all the samples in $\mathbf{\Gamma}$, we select the sample which has the smallest sum of distances in geometrical parameter space to all the other samples outside $\mathbf{\Gamma}$ with higher orders of transfer function. This selected sample will be the first sample whose order of transfer function needs to be increased by one. Let K represent the index of this selected sample, which is calculated as,

$$K = \arg\min_{k \in \mathbf{\Gamma}} \left\{ \sum_{l \in T_r, l \notin \mathbf{\Gamma}} \sum_{i=1}^{n_x} \left\| \frac{x_i^l - x_i^k}{\delta_i} \right\| \right\} \tag{4.6}$$

where δ_i is the smallest variance of x_i among all the samples. In (4.6), x_i^k and x_i^l represent the k^{th} and l^{th} samples of the i^{th} geometrical parameter, respectively. For the K^{th} sample, we choose one pole to be split into two overlapping poles as geometrical parameters change, according to the information of the nearest neighboring sample. The nearest neighboring sample (denoted as the L^{th} sample) is the one which has the shortest distance in geometrical parameter space to the K^{th} sample among all the samples whose orders of transfer function are equal to $N_K + 1$ [30]. Let m repre-

sent the index of the m^{th} pole in the K^{th} sample, where $m \in \{1, 2, ..., N_K\}$. Assume the m^{th} pole is to be split, we calculate the sum of differences D_m. D_m is defined as the total distances of pole movement along the imaginary axis as geometrical parameters change from the K^{th} sample to the L^{th} sample [30].

Here we first illustrate the selection of the pole to be split using the existing pole-residue tracking technique in Chapter 3 [30]. The information from this selection becomes the base, over which our proposed new pole-splitting technique is developed to address more challenging cases when training data become fewer. To determine which pole should be selected to be split, reference [30] compares different D_m, $m = 1, 2, ..., N_K$. The pole with the minimum D_m value is selected as the splitting pole [30]. Thus, the index of the selected pole is calculated as

$$M = \arg \min_{m \in \{1,2,...,N_K\}} \{D_m\} \tag{4.7}$$

In other words, the M^{th} pole of the K^{th} geometrical sample is selected to be split. By using (4.7), the poles move smoothly (continuously) instead of abruptly as geometrical parameters change.

Now, in the present chapter, we consider a more challenging case, i.e., fewer training data and large geometrical step size between the data (considering large geometrical range of variables). This case may lead to a situation where multiple sums of differences D_m are simultaneously very close to D_M. In this situation, it is difficult to distinguish the correct pole to be split from the other poles which should not be split. Thus, the pole-splitting may lead to non-unique choices using existing pole-residue tracking technique [30]. In this case (i.e., fewer data and large step size

between the data), it is difficult to decide which pole is the correct pole to be split.

To address this situation, we propose a new and advanced pole-residue tracking technique using derivative information of the poles to obtain reliable pole-splitting especially when we have few data. Let $\mathbf{\Phi}$ be an index set containing the indices of poles whose sum of differences D_m is close to D_M, defined as,

$$\mathbf{\Phi} = \left\{ m \left| \frac{(D_m - D_M)}{D_M} \leq \varepsilon \right. \right\} \tag{4.8}$$

where ε is a user defined threshold which decides whether D_m is close to D_M. Usually, ε is selected to be $0.1 - 0.2$. In order to distinguish between different poles (i.e., poles in the set $\mathbf{\Phi}$), which all appear almost equally qualified as the splitting pole using existing criteria in Chapter 3 [30], we consider the derivative information of the imaginary part of the poles. The derivative information can play a significant role in helping us identify the correct pole to be split from the other poles which should not be split. One of the poles in the K^{th} sample is eventually split into two poles in the L^{th} sample, as the geometrical parameters are changed from that in the K^{th} sample to that in the L^{th} sample. We use the derivative information to assess whether two neighboring poles in the L^{th} sample are moving together or apart from each other. Let Z_m represent the difference of the derivatives for all the geometrical parameters between the m^{th} and $(m+1)^{th}$ poles in the L^{th} sample, calculated as,

$$Z_m = \left(\frac{\partial \tilde{\beta}^L_{m+1}}{\partial \boldsymbol{x}^T} - \frac{\partial \tilde{\beta}^L_m}{\partial \boldsymbol{x}^T} \right) \left(\boldsymbol{x}^L - \boldsymbol{x}^K \right) \tag{4.9}$$

Larger positive values of Z_m means that the m^{th} and $(m+1)^{th}$ poles in the L^{th} sample are more likely to be split apart from the m^{th} pole in the K^{th} sample.

Therefore, we propose to select the $\widehat{M}^{th}$ pole in the K^{th} sample which has the maximum Z_m using the derivative information. The index of the selected pole is calculated as,

$$\widehat{M} = \arg\max_{m \in \mathbf{\Phi}} \{Z_m\} \tag{4.10}$$

Now, the $\widehat{M}^{th}$ pole will then be split into two new poles in the K^{th} sample. The real and imaginary values of these two new poles are defined to be equal to $\tilde{\alpha}^K_{\widehat{M}}$ and $\tilde{\beta}^K_{\widehat{M}}$, respectively. The $\widehat{M}^{th}$ residue corresponding to the selected $\widehat{M}^{th}$ pole should also be split into two separate residues. Each of these new residues are defined to be equal to half the value of the original residue. After splitting the $\widehat{M}^{th}$ pole, the number of elements in $\tilde{\boldsymbol{\alpha}}^K$, $\tilde{\boldsymbol{\beta}}^K$, $\tilde{\boldsymbol{\eta}}^K$ and $\tilde{\boldsymbol{\xi}}^K$ increases from N_K to $N_K + 1$. The real and imaginary parts of the poles of the K^{th} sample with the $\widehat{M}^{th}$ pole split, i.e., with the $\widehat{M}^{th}$ pole appearing twice, are expressed as,

$$\tilde{\boldsymbol{\alpha}}^K = \begin{bmatrix} \tilde{\alpha}^K_1 \ \tilde{\alpha}^K_2 \cdots \tilde{\alpha}^K_{\widehat{M}} \ \tilde{\alpha}^K_{\widehat{M}} \cdots \tilde{\alpha}^K_{N_K} \end{bmatrix}^T \tag{4.11}$$

$$\tilde{\boldsymbol{\beta}}^K = \begin{bmatrix} \tilde{\beta}^K_1 \ \tilde{\beta}^K_2 \cdots \tilde{\beta}^K_{\widehat{M}} \ \tilde{\beta}^K_{\widehat{M}} \cdots \tilde{\beta}^K_{N_K} \end{bmatrix}^T \tag{4.12}$$

The real and imaginary parts of the corresponding residues are expressed as,

$$\tilde{\boldsymbol{\eta}}^K = \begin{bmatrix} \tilde{\eta}^K_1 \ \tilde{\eta}^K_2 \cdots \dfrac{\tilde{\eta}^K_{\widehat{M}}}{2} \ \dfrac{\tilde{\eta}^K_{\widehat{M}}}{2} \cdots \tilde{\eta}^K_{N_K} \end{bmatrix}^T \tag{4.13}$$

$$\tilde{\boldsymbol{\xi}}^K = \begin{bmatrix} \tilde{\xi}^K_1 \ \tilde{\xi}^K_2 \cdots \dfrac{\tilde{\xi}^K_{\widehat{M}}}{2} \ \dfrac{\tilde{\xi}^K_{\widehat{M}}}{2} \cdots \tilde{\xi}^K_{N_K} \end{bmatrix}^T \tag{4.14}$$

The same splitting will also be applied to the derivative matrix $\boldsymbol{A}_k$, i.e., the size of the matrix will expend from $4N_k \times n_x$ to $4(N_k + 1) \times n_x$. The derivatives of the $\widehat{M}^{th}$

pole will be split into two new equal value derivatives as geometrical parameters change. The derivatives of the $\widehat{M}^{th}$ residue should also be split into two separate derivatives whose values are half of the original derivative value. After this process, the effective order of transfer function for the K^{th} sample is increased by one, i.e.,

$$N_K = N_K + 1 \tag{4.15}$$

This pole splitting is performed iteratively until the effective orders of transfer function of all the samples become equal to the maximum effective order N, i.e.,

$$N_k = N, \quad \forall k = 1, 2, ..., n_s \tag{4.16}$$

By using derivative information in pole-residue tracking technique, the pole splitting becomes more reliable, especially when the step sizes between data are large. The final data of poles and residues reformatted using the proposed method are continuous with respect to the changes in geometrical parameters. The data obtained after pole-residue tracking process can be further used in parametric modeling using combined neural networks and pole-residue-based transfer functions.

4.3 Application Example

4.3.1 Illustration of Pole-Residue Tracking Technique Using a Coupled-Line Filter Example

The proposed advanced pole-residue tracking technique using EM sensitivity analysis of a coupled-line filter example with one-dimension (one geometrical parameter) is illustrated in this section. Fig. 4.1 shows the structure for the EM simulation of

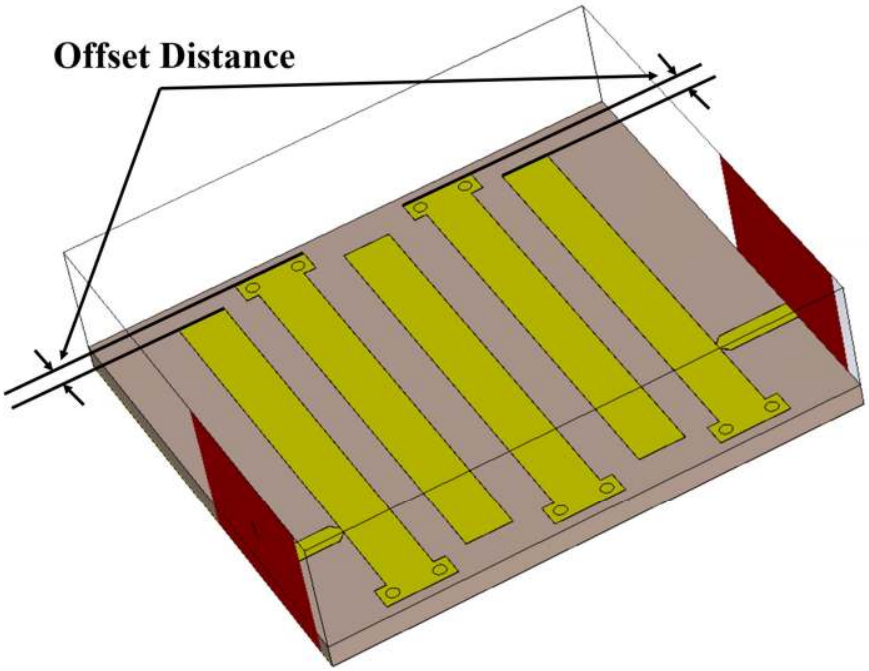


Figure 4.1: The structure for the EM simulation of the coupled-line filter, where the geometrical variable used in this example is the offset distance from the ends of the two lines near the ports to the corresponding fringes. In this example, the number of poles changes as the value of the offset distance changes.

the coupled-line filter. The geometrical parameter used in this example is the offset distance from the ends of the two lines near the ports to the corresponding fringes. The number of effective poles changes from five to six as the geometrical parameter (offset distance) changes using vector fitting technique.

In this example, two of the poles (one having the largest value of imaginary part, another having the smallest value of imaginary part) in the K^{th} sample are clearly matching with two of the poles in the L^{th} sample. Therefore these poles do not need to be considered as candidates for pole splitting. The remaining three poles in the K^{th} sample and four poles in the L^{th} sample are considered for pole splitting as

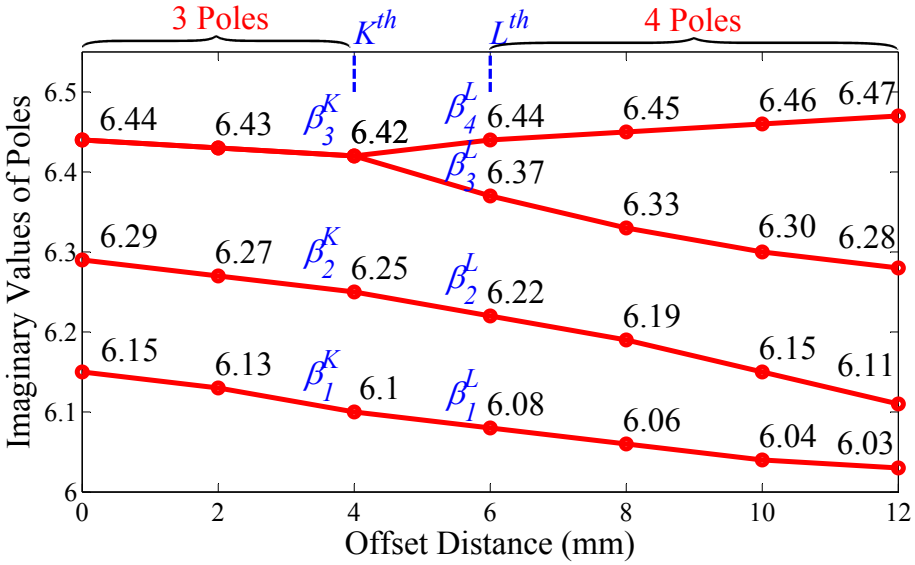


Figure 4.2: In order to track how the three poles change to four poles, we derive a pole splitting technique where one of the poles is split into two new poles as geometrical variables change. When the geometrical variable changes in small step sizes, the correct pole is chosen to be split using the existing pole-residue tracking technique [30].

illustrated in Fig. 4.2. As shown in Fig. 4.2, when the offset distance is equal to 4 mm (i.e., the K^{th} sample), the number of poles is three. When the offset distance is changed to 6 mm (i.e., the L^{th} sample), the number of poles is changed to four.

In order to track how the three poles change to four poles, we derive a pole splitting technique where one of the poles is split into two new poles as geometrical variables change from the K^{th} sample to the L^{th} sample. In other words, one of the three poles in the K^{th} sample needs to be split in order to make the number of poles in the K^{th} sample to be the same as that in the L^{th} sample. Fig. 4.2 is used to illustrate how pole splitting is performed in the existing pole-residue tracking

technique [30]. The sum of differences D_m for the m^{th} pole ($m = 1, 2, 3$) in the K^{th} sample is calculated in the example, i.e., $D_1 = 0.28$, $D_2 = 0.19$ and $D_3 = 0.12$. The minimum sum of differences is D_3, therefore $M = 3$. This result illustrates that the imaginary values of the third and the fourth poles in the L^{th} sample are close to the value of the third pole in the K^{th} sample. Therefore, the third pole in the K^{th} sample is selected to be split, i.e., $M = 3$. As shown in Fig. 4.2, when the geometrical step sizes are small, the pole which needs to be split can be identified correctly using the existing pole-residue tracking technique in Chapter 3 [30].

4.3.2 Advanced Pole-Residue Tracking Technique Using Sensitivity Analysis of The Coupled-Line Filter Example

Now, in the present sub-section, we consider a more challenging case, i.e., fewer training data and large geometrical step size between the data (considering large geometrical range of variables). This situation is shown in Fig. 4.3. The geometrical parameter (offset distance) of the example is changed by a larger step size from 4 mm (i.e., the K^{th} sample) to 8 mm (i.e., the L^{th} sample). The sum of differences D_m for the m^{th} pole ($m = 1, 2, 3$) in the K^{th} sample is calculated to be $D_1 = 0.24$, $D_2 = 0.21$ and $D_3 = 0.22$. The minimum sum of differences is $D_M = D_2 = 0.21$. In this situation, the value of each D_m is very similar to D_M. When pole splitting is performed using existing pole-residue tracking technique [30], each of the three poles qualifies to be chosen for splitting as the geometrical parameters change, because D_1, D_2 and D_3 are very similar. As shown in Fig. 4.3, the dash lines illustrate the possibilities of pole splitting using existing pole-residue tracking technique. In this

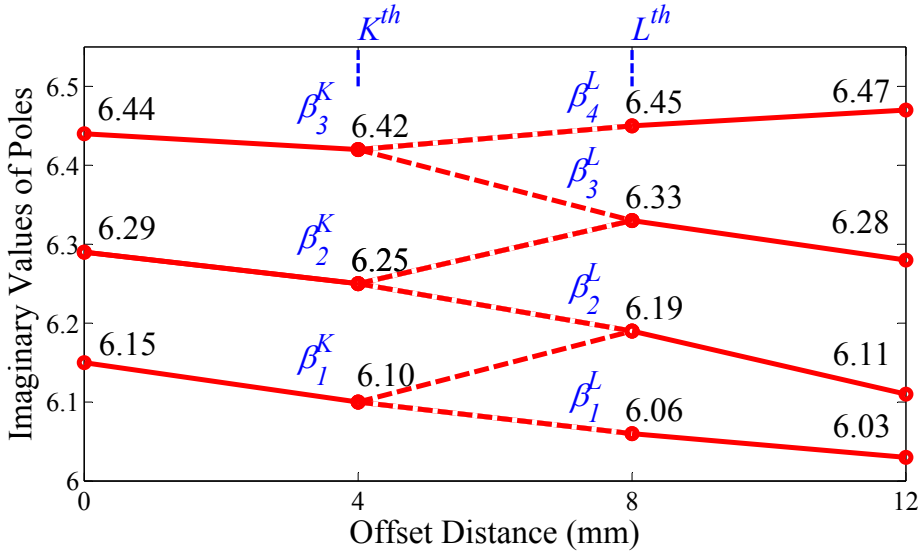


Figure 4.3: When the geometrical variable changes in large step sizes, there are multiple possibilities of pole splitting using existing pole-residue tracking technique [30] because of similar sums of differences.

case (i.e., fewer data and large step size between the data), it is difficult to decide which pole is the correct pole to be split.

Using (4.10), the pole which needs to be split is identified correctly, shown in Fig. 4.4. As illustrated in Fig. 4.4, we calculate the difference of the derivatives Z_m for the m^{th} pole ($m = 1, 2, 3$) in the example, i.e., $Z_1 = -0.0564$, $Z_2 = 0.0376$, $Z_3 = 0.0712$. The difference of derivatives Z_3 between the third and the fourth poles in the L^{th} sample has the maximum value among all the differences of derivatives between neighboring poles in the L^{th} sample. This means the third and fourth poles in the L^{th} sample are most likely to be split apart from the third pole in the K^{th} sample. Using the proposed advanced pole-residue tracking technique, the third

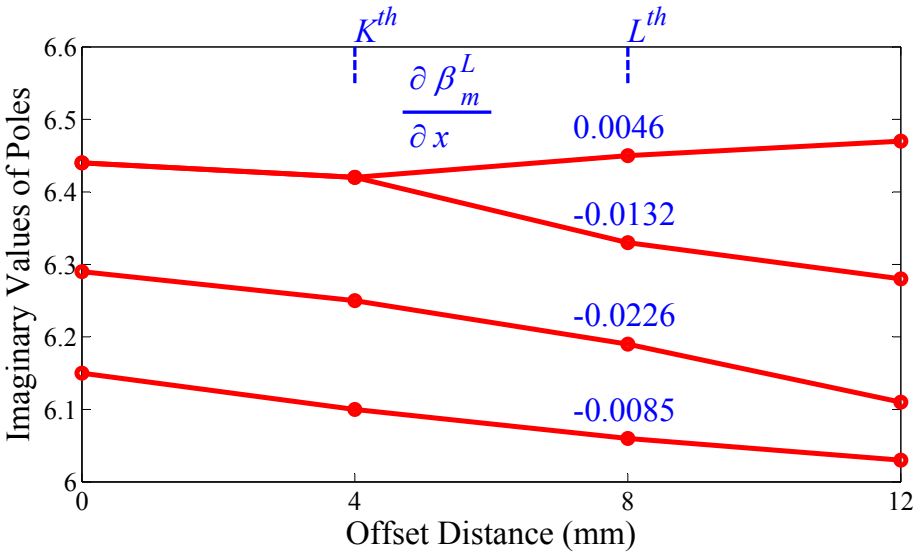


Figure 4.4: When the geometrical variable changes in large step sizes, the correct pole is chosen to be split using the proposed pole-residue tracking technique with sensitivity information.

pole with value 6.42 in the K^{th} sample is the one which needs to be split, i.e., $\widehat{M} = 3$. Thus, using the proposed pole-residue tracking technique with derivative information, the correct pole is selected to be split as the geometrical parameters change.

4.4 Conclusion

In this chapter, a new pole-residue tracking technique using EM sensitivity analysis has been proposed. This technique is proposed to use the derivative information to obtain more reliable pole splitting than the existing pole-residue tracking technique. This pole-residue tracking technique allows the model to bridge the differences of

the orders of transfer function over different regions of the geometrical parameters, and ultimately form smooth and continuous functions between the pole/residues and the geometrical variables. The proposed technique can solve the challenges of tracking pole splitting when training data are limited.

Chapter 5

Parametric Modeling Using Neuo-Transfer Functions with EM Sensitivity Analysis

This chapter proposes a pole-residue-based adjoint neuro-transfer function (neuro-TF) technique with EM sensitivity analysis for parametric modeling of EM behavior of microwave components with respect to changes in geometrical parameters. The purpose is to increase model accuracy by utilizing EM sensitivity information and to speedup model development by reducing the number of training data required for developing the model. The proposed parametric model consists of original and adjoint pole-residue based neuro-TF models. New formulations are derived for calculating the second order derivatives for training the adjoint pole-residue based neuro-TF model. The proposed method utilize the advanced pole-residue tracking technique using EM sensitivity analysis to track the splitting of poles as geometrical parameters change. By exploiting the sensitivity information, the proposed technique can speed up the model development process over the existing pole-residue

parametric modeling method which does not use sensitivity analysis.

5.1 Introduction

As a further development of parametric modeling, sensitivity analysis has been used to speed up the model development process by utilizing EM sensitivity information to reduce the number of training data required for developing the model. In the reference [56], a method using adjoint neural network is presented to perform the sensitivity analysis in ANN modeling. A parametric modeling method using sensitivity-analysis-based adjoint neural network technique for solving the passive microwave components is described in the reference [14]. The techniques in the reference [14] and [56] can train the parametric model with both the input-output data and the derivative data of EM problems simultaneously to obtain a robust model. To calculate these EM derivative data, EM field based adjoint sensitivity analysis techniques, e.g., the reference [15], can be used. With sensitivity information (EM derivative data), we can retain the model accuracy using less data for training the model.

This chapter presents a further advance over the work of [30]. A new sensitivity-analysis-based neuro-transfer function (neuro-TF) modeling technique using transfer functions in pole/residue format is proposed for parametric modeling of microwave components. The proposed parametric model is trained using the input-output behavior and the sensitivity analysis information generated from EM simulation simultaneously. The neuro-transfer function in pole/residue format is used in this chapter. We propose new formulations to calculate the second-order derivatives

required for developing the pole-residue-based adjoint neuro-TF model. The proposed method utilize the advanced pole-residue tracking technique using sensitivity information to solve the challenge of pole splitting especially when the amount of training data are reduced and/or the geometrical step sizes between the data samples are enlarged. By exploiting sensitivity information for training, the proposed modeling method can obtain accurate parametric models with fewer training data (or obtain better accuracy with the same amount of training data), compared to the previous neuro-TF modeling method without using sensitivity [30].

5.2 Proposed Sensitivity-Analysis-Based Neuro-TF Modeling Technique

The proposed sensitivity-analysis-based neuro-TF model is shown in Fig. 5.1. It consists of two parts: the original neuro-TF model and the adjoint neuro-TF model. Let $\boldsymbol{x}$ represent the input vector of the original neuro-TF model such as geometrical parameters of microwave passive components. Let $\boldsymbol{y}$ represent the outputs of the original neuro-TF model such as S-parameters. The adjoint neuro-TF model shares the same inputs $\boldsymbol{x}$ of the original neuro-TF model. The outputs of the adjoint neuro-TF model are the derivatives of the original neuro-TF model outputs $\boldsymbol{y}$ with respect to the inputs $\boldsymbol{x}$, denoted as $d\boldsymbol{y}/d\boldsymbol{x}$. The outputs of the overall proposed model consists of both the outputs $\boldsymbol{y}$ of the original neuro-TF model and the outputs $d\boldsymbol{y}/d\boldsymbol{x}$ of the adjoint neuro-TF model.

The transfer functions in pole/residue format are used in the proposed sensitivity-analysis-based neuro-TF model. Poles and residues of the transfer function will

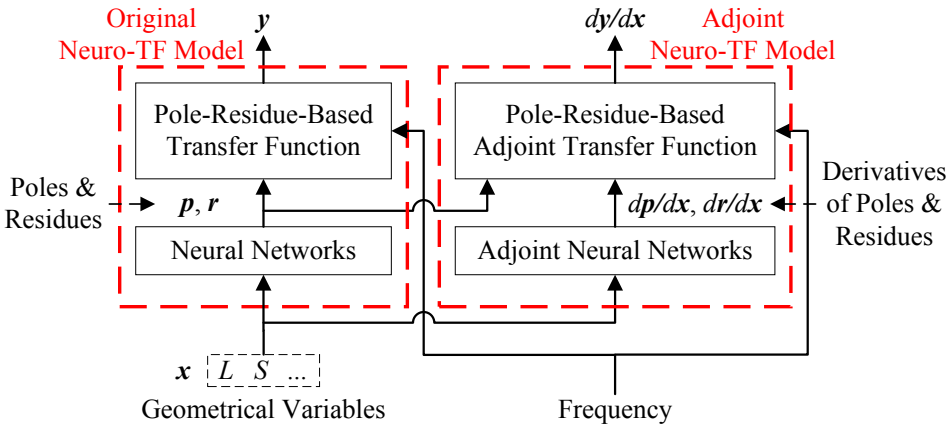


Figure 5.1: The structure of the proposed sensitivity-analysis-based neuro-TF model with transfer functions in pole/residue format. The proposed model contains the original neuro-TF model and the adjoint neuro-TF model, where $\boldsymbol{x}$ represents the geometrical variables; $\boldsymbol{y}$ represents the outputs of the transfer function; $d\boldsymbol{y}/d\boldsymbol{x}$ represents the derivatives of $\boldsymbol{y}$ with respect to $\boldsymbol{x}$.

change as the geometrical parameters change. Because the relationship between the pole/residues and the geometrical parameters is non-linear and unknown, neural networks are used to represent the relationship. The neural network used in the original neuro-TF model is called the original neural network. Consequently, adjoint neural networks [14] are used in the adjoint neuro-TF model to map the geometrical parameters onto the derivatives of pole/residues.

The outputs $\boldsymbol{y}$ contains the real/imaginary parts of transfer function response H, which is formulated in pole/residue format [30],

$$H(s, \boldsymbol{x}, \boldsymbol{w}) = \sum_{i=1}^{N} \frac{r_i(\boldsymbol{x}, \boldsymbol{w})}{s - p_i(\boldsymbol{x}, \boldsymbol{w})} + \sum_{i=1}^{N} \frac{r_i^*(\boldsymbol{x}, \boldsymbol{w})}{s - p_i^*(\boldsymbol{x}, \boldsymbol{w})} \tag{5.1}$$

where N represents the effective order of transfer function. p_i and r_i represent the effective poles and residues, respectively. The superscript "$*$" means complex con-

jugate. s represents the frequency in Laplace domain. $\boldsymbol{w}$ represents the weighting parameters in the neural networks. Let $\boldsymbol{p}$ and $\boldsymbol{r}$ represent vectors containing all the poles and residues, respectively. Separate neural networks are used to represent the real and imaginary parts of the pole/residues in order to maintain real value neural network structures, defined as

$$\begin{aligned} \boldsymbol{p}(\boldsymbol{x}, \boldsymbol{w}) &= \boldsymbol{\alpha}(\boldsymbol{x}, \boldsymbol{w}) + j\boldsymbol{\beta}(\boldsymbol{x}, \boldsymbol{w}) \\ \boldsymbol{r}(\boldsymbol{x}, \boldsymbol{w}) &= \boldsymbol{\eta}(\boldsymbol{x}, \boldsymbol{w}) + j\boldsymbol{\xi}(\boldsymbol{x}, \boldsymbol{w}) \end{aligned} \tag{5.2}$$

where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ represent the vectors containing the real and imaginary parts of the poles $\boldsymbol{p}$, respectively, while $\boldsymbol{\eta}$ and $\boldsymbol{\xi}$ represent the vectors containing the real and imaginary parts of the residues $\boldsymbol{r}$, respectively. $\boldsymbol{w}$ represents the weighting parameters in the neural networks. $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\eta}$, and $\boldsymbol{\xi}$ represent the functions of neural networks with respect to $\boldsymbol{x}$ and $\boldsymbol{w}$.

5.2.1 Pole-Residue Extraction and Derivative Data Calculation

In the proposed method, pole-residue-based transfer function formulation is used for the sensitivity-analysis-based neuro-TF model. Firstly, we perform parameter extraction to obtain the data for preliminary training of the neural networks $\boldsymbol{\alpha}(\boldsymbol{x}, \boldsymbol{w})$, $\boldsymbol{\beta}(\boldsymbol{x}, \boldsymbol{w})$, $\boldsymbol{\eta}(\boldsymbol{x}, \boldsymbol{w})$, and $\boldsymbol{\xi}(\boldsymbol{x}, \boldsymbol{w})$. The parameter extraction process consists of pole-residue extraction and pole-residue derivative calculation. Pole-residue extraction is to obtain the data of poles and residues with respect to geometrical parameters $\boldsymbol{x}$ for training the original neural networks. Pole-residue derivative calculation is to obtain the data of derivatives of poles and residues with respect to $\boldsymbol{x}$ for training

the adjoint neural networks.

Different samples of geometrical parameters, which are also called as training samples, are generated with respect to the inputs $\boldsymbol{x}$ containing all the geometrical parameters except frequency which is a separate variable swept by the EM simulator during data generation. Let $\boldsymbol{d}$ represent the data of the outputs generated from EM simulations (e.g., S-parameters). Let $\boldsymbol{d}'$ represent the derivative data of the outputs with respect to geometrical parameters from EM sensitivity analysis. EM derivative data $\boldsymbol{d}'$ is generated using EM sensitivity analysis simultaneously with the generation of EM output data $\boldsymbol{d}$ in the EM simulator. Let T_r represent the index set of the training samples of different values of geometrical parameters, i.e., $T_r = \{1, 2, ..., n_s\}$, where n_s is the total number of the training samples of geometrical parameters.

Pole-residue extraction is performed using a vector fitting process [121]. We use vector fitting to obtain a group of effective poles and residues from the EM output data $\boldsymbol{d}$ versus frequency, using the minimum order of transfer function for each geometrical sample. With minimum order of transfer function, we can avoid non-unique and arbitrary numerical solutions of pole/residues from the vector fitting process. However, by doing so, the orders of transfer function may vary among different geometrical samples. Let N_k represent the effective order of transfer function for the k^{th} geometrical sample, where k represents the index of the geometrical samples, i.e., $k \in T_r$. N_k is initially equal to the minimum effective order of transfer function for the k^{th} sample determined from the vector fitting process.

To make the output data for both original neural network and adjoint neural

network real values, the real and imaginary parts of the effective poles and residues are used in our formulation to form the transfer function. Let $\tilde{\boldsymbol{\alpha}}$ and $\tilde{\boldsymbol{\beta}}$ represent the real and imaginary parts of the poles of the transfer function whose order may vary among different geometrical samples. Let $\tilde{\boldsymbol{\eta}}$ and $\tilde{\boldsymbol{\xi}}$ represent the real and imaginary parts of the corresponding residues, respectively. Let $\tilde{\boldsymbol{c}}$ represent a vector containing these real and imaginary parts of poles and residues, defined as

$$\tilde{\boldsymbol{c}} = \left[\tilde{\boldsymbol{\alpha}}^T \ \tilde{\boldsymbol{\beta}}^T \ \tilde{\boldsymbol{\eta}}^T \ \tilde{\boldsymbol{\xi}}^T\right]^T \tag{5.3}$$

Let $\tilde{\boldsymbol{c}}_k$ represent $\tilde{\boldsymbol{c}}$ at a particular sample, i.e., the k^{th} sample, containing $\tilde{\boldsymbol{\alpha}}^k$, $\tilde{\boldsymbol{\beta}}^k$, $\tilde{\boldsymbol{\eta}}^k$ and $\tilde{\boldsymbol{\xi}}^k$, which represent real and imaginary parts of poles and residues at the k^{th} sample.

After pole-residue extraction, we perform the pole-residue derivative calculation. Let Ω represent the index set of frequencies, i.e., $\Omega = \{1, 2, ..., n_f\}$, where n_f is the total number of frequencies. The frequency response H is expressed as a function of pole/residues and frequency s_q, where q represents the index of frequencies, i.e., $q \in \Omega$. Let H_q^k represent the transfer function response at the q^{th} frequency for the k^{th} sample. The response of the adjoint neuro-TF output at the q^{th} frequency for the k^{th} sample is calculated as,

$$\frac{\partial H_q^k}{\partial \boldsymbol{x}} = \frac{\partial \tilde{\boldsymbol{c}}_k^T}{\partial \boldsymbol{x}} \frac{\partial H_q^k}{\partial \tilde{\boldsymbol{c}}} \tag{5.4}$$

From (5.4), we need to first obtain the derivative data of EM outputs with respect to $\boldsymbol{x}$ and derivative data of EM outputs with respect to poles and residues. Subsequently, the derivative data of poles and residues with respect to geometrical

parameters $\boldsymbol{x}$ need to be calculated.

First, we obtain the matrix of derivative data of EM outputs with respect to $\boldsymbol{x}$ for all the training samples. Let $\boldsymbol{g}_q^k$ represent the derivative data of the EM response H_q^k with respect to geometrical parameters $\boldsymbol{x}$ at the q^{th} frequency of the k^{th} training sample, defined as,

$$\boldsymbol{g}_q^k = \frac{\partial H_q^k}{\partial \boldsymbol{x}} \tag{5.5}$$

Let $\boldsymbol{G}_k$ represent the derivative data of the EM outputs with respect to $\boldsymbol{x}$ for the k^{th} training sample, containing both the real and imaginary parts of $\boldsymbol{g}_q^k$ at all the frequency points, defined as

$$\begin{aligned}\boldsymbol{G}_k =&[\mathrm{Re}(\boldsymbol{g}_1^k)\ \mathrm{Re}(\boldsymbol{g}_2^k)\cdots\mathrm{Re}(\boldsymbol{g}_{n_f}^k)\\ &\mathrm{Im}(\boldsymbol{g}_1^k)\ \mathrm{Im}(\boldsymbol{g}_2^k)\cdots\mathrm{Im}(\boldsymbol{g}_{n_f}^k)]^T\end{aligned} \tag{5.6}$$

$\boldsymbol{G}_k$ is a $2n_f \times n_x$ matrix formed from EM derivative data $\boldsymbol{d}'$, where n_x represents the total number of the geometrical parameters in vector $\boldsymbol{x}$.

Then, we need to calculate the derivative data of EM outputs with respect to poles and residues. Let $\boldsymbol{f}_q^k$ represent a vector which contains the derivatives of the transfer function response with respect to the real and imaginary parts of all the poles and residues at the q^{th} frequency for the k^{th} sample, defined as

$$\boldsymbol{f}_q^k = \left[(\frac{\partial H_q^k}{\partial \tilde{\boldsymbol{\alpha}}})^T\ (\frac{\partial H_q^k}{\partial \tilde{\boldsymbol{\beta}}})^T\ (\frac{\partial H_q^k}{\partial \tilde{\boldsymbol{\eta}}})^T\ (\frac{\partial H_q^k}{\partial \tilde{\boldsymbol{\xi}}})^T\right]^T \tag{5.7}$$

where the elements in $\boldsymbol{f}_q^k$ are calculated as

$$\frac{\partial H_q^k}{\partial \tilde{\alpha}_i} = \frac{r_i^k}{(s_q - p_i^k)^2} + \frac{r_i^{k*}}{(s_q - p_i^{k*})^2} \tag{5.8}$$

$$\frac{\partial H_q^k}{\partial \tilde{\beta}_i} = \frac{jr_i^k}{(s_q - p_i^k)^2} - \frac{jr_i^{k*}}{(s_q - p_i^{k*})^2} \tag{5.9}$$

$$\frac{\partial H_q^k}{\partial \tilde{\eta}_i} = \frac{1}{s_q - p_i^k} + \frac{1}{s_q - p_i^{k*}} \tag{5.10}$$

$$\frac{\partial H_q^k}{\partial \tilde{\xi}_i} = \frac{j}{s_q - p_i^k} - \frac{j}{s_q - p_i^{k*}} \tag{5.11}$$

where i represents the index of pole/residues, i.e., $i \in \{1, 2, ..., N_k\}$, and k represents the index of the samples, i.e., $k \in \{1, 2, ..., n_s\}$. Let $\boldsymbol{F}_k$ represent the derivative data of the EM outputs with respect to the real and imaginary parts of poles and residues for the k^{th} training sample, containing the real and imaginary parts of $\boldsymbol{f}_q^k$ at all the frequency points. $\boldsymbol{F}_k$ is a $2n_f \times 4N_k$ matrix, defined as

$$\begin{aligned} \boldsymbol{F}_k = &[\mathrm{Re}(\boldsymbol{f}_1^k)\ \mathrm{Re}(\boldsymbol{f}_2^k) \cdots \mathrm{Re}(\boldsymbol{f}_{n_f}^k) \\ &\mathrm{Im}(\boldsymbol{f}_1^k)\ \mathrm{Im}(\boldsymbol{f}_2^k) \cdots \mathrm{Im}(\boldsymbol{f}_{n_f}^k)]^T \end{aligned} \tag{5.12}$$

Let $\boldsymbol{A}_k$ represent the data matrix for the derivatives of poles and residues with respect to geometrical parameters $\boldsymbol{x}$ for the k^{th} sample. $\boldsymbol{A}_k$ is a $4N_k \times n_x$ matrix, which is expressed as,

$$\boldsymbol{A}_k = \left[(\frac{\partial \tilde{\boldsymbol{\alpha}}^k}{\partial \boldsymbol{x}^T})^T\ (\frac{\partial \tilde{\boldsymbol{\beta}}^k}{\partial \boldsymbol{x}^T})^T\ (\frac{\partial \tilde{\boldsymbol{\eta}}^k}{\partial \boldsymbol{x}^T})^T\ (\frac{\partial \tilde{\boldsymbol{\xi}}^k}{\partial \boldsymbol{x}^T})^T \right]^T \tag{5.13}$$

After matrices $\boldsymbol{G}_k$ and $\boldsymbol{F}_k$ are obtained, we can calculate $\boldsymbol{A}_k$. From (5.4), the equation for solving $\boldsymbol{A}_k$ for the k^{th} sample is formulated as,

$$\boldsymbol{F}_k \boldsymbol{A}_k = \boldsymbol{G}_k \tag{5.14}$$

Because $\boldsymbol{F}_k$ is not a square matrix, in order to calculate $\boldsymbol{A}_k$, we reformulate (5.14)

into the following equation as

$$\boldsymbol{A}_k = (\boldsymbol{F}_k^T \boldsymbol{F}_k)^{-1} \boldsymbol{F}_k^T \boldsymbol{G}_k \tag{5.15}$$

During the calculation of $\tilde{\boldsymbol{c}}_k$ and $\boldsymbol{A}_k$, we use minimum orders of the transfer function for each geometrical sample to avoid non-unique and arbitrary numerical solutions of pole/residues from the vector fitting process. However, that may result in varied orders of the transfer function for different geometrical samples. The varied orders can lead to abrupt changes in values of poles and residues, and/or abrupt changes in the function structure of poles and residues as geometrical parameters change. These abrupt changes will lead to the discontinuity of pole/residues with respect to changes in geometrical parameters.

The advanced pole-residue tracking technique using EM sensitivity analysis is utilized in this chapter to overcome the discontinuity problem of pole/residues. Let $\hat{\boldsymbol{c}}_k$ and $\hat{\boldsymbol{A}}_k$ represent the reformatted data of pole/residues and their derivatives with respect to $\boldsymbol{x}$, respectively, using the proposed pole-residue tracking technique. $\hat{\boldsymbol{c}}_k$ and $\hat{\boldsymbol{A}}_k$ are expressed as,

$$\hat{\boldsymbol{c}}_k = \left[\hat{\alpha}_1^k \cdots \hat{\alpha}_N^k \;\; \hat{\beta}_1^k \cdots \hat{\beta}_N^k \;\; \hat{\eta}_1^k \cdots \hat{\eta}_N^k \;\; \hat{\xi}_1^k \cdots \hat{\xi}_N^k\right]^T \tag{5.16}$$

$$\hat{\boldsymbol{A}}_k = \left[\frac{\partial \hat{\alpha}_1^k}{\partial \boldsymbol{x}} \cdots \frac{\partial \hat{\alpha}_N^k}{\partial \boldsymbol{x}} \;\; \frac{\partial \hat{\beta}_1^k}{\partial \boldsymbol{x}} \cdots \frac{\partial \hat{\beta}_N^k}{\partial \boldsymbol{x}} \;\; \frac{\partial \hat{\eta}_1^k}{\partial \boldsymbol{x}} \cdots \frac{\partial \hat{\eta}_N^k}{\partial \boldsymbol{x}} \;\; \frac{\partial \hat{\xi}_1^k}{\partial \boldsymbol{x}} \cdots \frac{\partial \hat{\xi}_N^k}{\partial \boldsymbol{x}}\right]^T \tag{5.17}$$

Notice that the size of the vector $\hat{\boldsymbol{c}}_k$ is a constant being $4N$ for all samples, $k = 1, 2, ..., n_s$. The matrix $\hat{\boldsymbol{A}}_k$ has a constant size of $4N \times n_x$.

5.2.2 Preliminary Training of the Sensitivity-Analysis-Based Neuro-TF Model

The sensitivity-analysis-based neuro-TF model is trained with both EM simulation data and EM sensitivity data simultaneously. With the same amount of training data samples, we can make the model more accurate by training with the sensitivity information than that by training without sensitivity information. Alternatively, we can also maintain the model accuracy with less training data samples by using the sensitivity information. A two stage training process is proposed in the sensitivity-analysis-based modeling technique. In the first stage, we perform a preliminary training. In the second stage, a model refinement is proposed to obtain an accurate model.

In the first stage, the preliminary training of the original neural network and adjoint neural network are performed simultaneously. The original neural network represents the mapping between the pole/residues and geometrical parameters $\boldsymbol{x}$. The adjoint neural network represents the mapping between the derivatives of pole/residues and geometrical parameters $\boldsymbol{x}$. The training data for the original neural networks and adjoint neural networks are $(\boldsymbol{x}_k,\ \hat{\boldsymbol{c}}_k)$ and $(\boldsymbol{x}_k,\ \hat{\boldsymbol{A}}_k)$, respectively, $k \in T_r$. $\hat{\boldsymbol{c}}_k$ and $\hat{\boldsymbol{A}}_k$ are obtained using the proposed pole-residue tracking technique, i.e., (5.16) and (5.17), respectively.

Let $\boldsymbol{c}(\boldsymbol{x}, \boldsymbol{w})$ represent a vector containing all the neural network outputs, i.e.,

$\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\eta}$, and $\boldsymbol{\xi}$. The error function for this training process is formulated as

$$\begin{aligned} E_{Pre}(\boldsymbol{w}) = & \frac{1}{2n_s}\sum_{k=1}^{n_s}\|\boldsymbol{c}(\boldsymbol{x}_k,\boldsymbol{w})-\hat{\boldsymbol{c}}_k\|^2 \\ & + \frac{1}{2n_s}\sum_{k=1}^{n_s}\left\|\frac{\partial \boldsymbol{c}(\boldsymbol{x}_k,\boldsymbol{w})}{\partial \boldsymbol{x}^T}-\hat{\boldsymbol{A}}_k\right\|_F^2 \end{aligned} \tag{5.18}$$

where n_s is the number of training samples of geometrical parameters. $\|\cdot\|$ and $\|\cdot\|_F$ represent L_2 norm and Frobenius norm, respectively.

Here we provide a comparison of the three symbols $\tilde{\boldsymbol{c}}_k$, $\hat{\boldsymbol{c}}_k$ and $\boldsymbol{c}$. All these three symbols are vectors containing real and imaginary parts of poles and residues. However, their meanings are different. The size of $\tilde{\boldsymbol{c}}_k$ for the k^{th} sample is $4N_k$, which may vary from sample to sample. The size of $\hat{\boldsymbol{c}}_k$ and $\boldsymbol{c}_k$ are both constant being $4N$. $\tilde{\boldsymbol{c}}_k$ represents the data of pole/residues whose size may change during the pole splitting process. $\hat{\boldsymbol{c}}_k$ represents the data of pole/residues for neural network training, which is equal to $\tilde{\boldsymbol{c}}_k$ after the last iteration of the pole splitting process. $\boldsymbol{c}$ represents the vector of the neural network functions, which is trained using data $\hat{\boldsymbol{c}}_k$.

5.2.3 Refinement Training of the Sensitivity-Analysis-Based Neuro-TF Model

A refinement training is proposed to further refine the overall model after the first-stage training. This is called the second-stage training. The training data for the second stage are ($\boldsymbol{x}_k$, $\boldsymbol{d}_k$) and ($\boldsymbol{x}_k$, $\boldsymbol{d}'_k$), $k \in T_r$, i.e., samples of geometrical parameters as model inputs and EM responses (i.e., S-parameters) and derivatives of EM responses with respect to geometrical parameters as model outputs. Fig. 5.2 shows

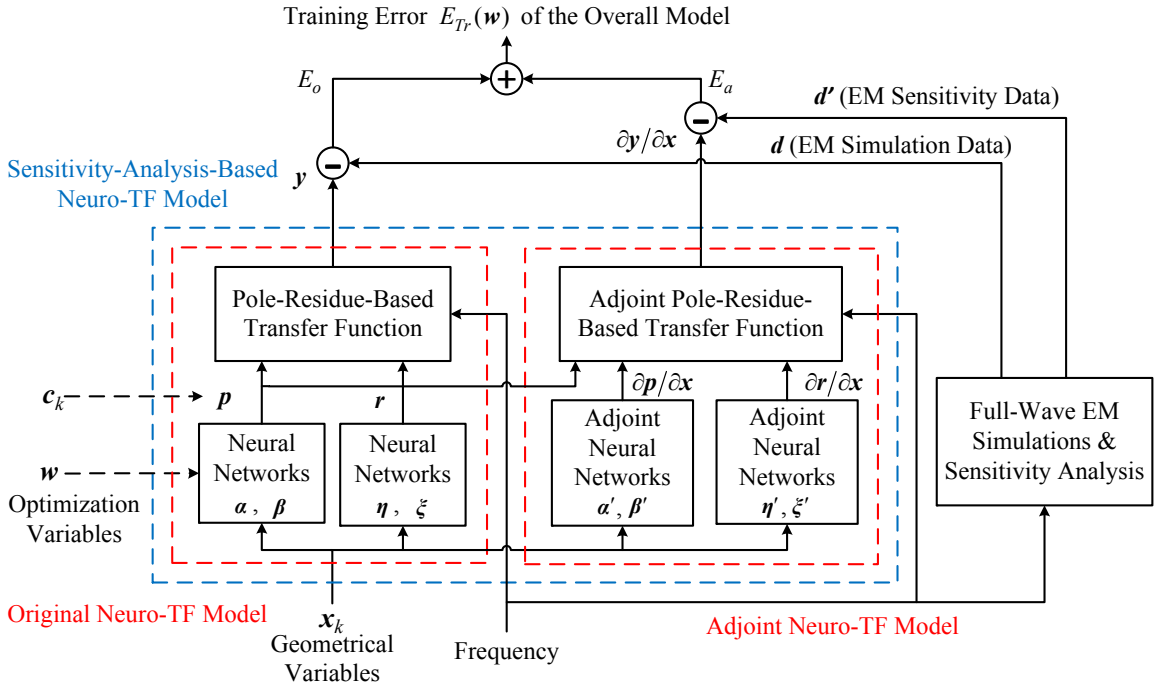


Figure 5.2: The mechanism for the refinement training process of the overall sensitivity-analysis-based neuro-TF model. The objective is to minimize the training errors of both the original and adjoint neuro-TF model simultaneously, i.e., the model outputs and model derivatives are trained to match the EM simulation data and the EM derivative data, respectively. The variables of this training process are the weighting parameters in neural networks $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\eta}$ and $\boldsymbol{\xi}$. After training, the original neuro-TF model is the final model ready to be used for high-level design.

the refinement training mechanism of the overall sensitivity-analysis-based neuro-TF model. The original and adjoint neuro-TF model are trained simultaneously during the proposed model refinement process. During this process, not only the output of the neuro-TF model is trained to be equal to the output data, but also the derivatives of the output w.r.t geometrical parameters are trained to be equal

to the derivative data. The error function of this training process is defined as

$$\begin{aligned} E_{Tr}(\boldsymbol{w}) &= E_{orig}(\boldsymbol{w}) + E_{adj}(\boldsymbol{w}) \\ &= \frac{1}{2n_s}\sum_{k=1}^{n_s}\sum_{j=1}^{n_y} a_j \|y_j(\boldsymbol{x}_k, \boldsymbol{w}) - d_{k,j}\|^2 \\ &\quad + \frac{1}{2n_s}\sum_{k=1}^{n_s}\sum_{j=1}^{n_y}\sum_{i=1}^{n_x} b_{j,i}\left\|\frac{\partial y_j(\boldsymbol{x}_k, \boldsymbol{w})}{\partial x_i} - d'_{k,j,i}\right\|^2 \end{aligned} \tag{5.19}$$

where E_{orig} represents the training error of the original neuro-TF models. E_{adj} represents the training error of the adjoint neuro-TF models. n_s represents the number of training samples of geometrical parameters. n_y represents the number of outputs in vector $\boldsymbol{y}$. n_x represents the number of the geometrical parameters in vector $\boldsymbol{x}$. $d_{k,j}$ represents the training data of EM evaluations of the j^{th} output for the k^{th} sample. $d'_{k,j,i}$ represents the training data of EM derivatives of the j^{th} output with respect to the i^{th} input for the k^{th} sample. a_j and $b_{j,i}$ represent the weighting parameters for the error functions of original and adjoint neuro-TF model, respectively. a_j and $b_{j,i}$ are adjustable during the model refinement process to maintain the robustness of the training of the overall model.

When the training error is less than a user defined error criteria, the training process terminates. An independent set of data which has not been used in training process is used for testing the accuracy of the trained model. The error between the original neuro-TF model responses and the testing data is defined as the testing error E_{Te}. We define the threshold error (i.e., user defined error criteria) for the original neuro-TF model to be E_t. The model refinement training process terminates when the testing error E_{Te} becomes less than the threshold E_t. After the model

refinement, the original neuro-TF model can be further used in design optimization. Fig. 5.3 shows a flowchart which illustrates the entire model development process of the proposed sensitivity-analysis-based neuro-TF model.

5.2.4 Derivative Calculation for Training the Proposed Sensitivity-Analysis-Based Neuro-TF Model

During the neuro-TF model training process in Chapter 3 [30], only the first-order derivative is needed. However, more derivative information will be required for training the proposed model. The error function of the proposed training in (5.19) contains the error of outputs E_{orig} and the error of the first order derivatives E_{adj}. Because the error function in (5.19) has the first order derivatives, the gradient of the error function means the second order derivatives, i.e., the second order derivatives of model outputs $\boldsymbol{y}$ with respect to model inputs $\boldsymbol{x}$ and neural network internal weights $\boldsymbol{w}$ are needed to perform the gradient-based training. In this chapter, we derive this second order derivatives.

The first order derivatives of $\boldsymbol{y}$ with respect to $\boldsymbol{w}$ are calculated similarly to [30] as,

$$\frac{\partial H(s,\boldsymbol{x},\boldsymbol{w})}{\partial \boldsymbol{w}} = \frac{\partial \boldsymbol{c}^T(\boldsymbol{x},\boldsymbol{w})}{\partial \boldsymbol{w}} \frac{\partial H(s,\boldsymbol{x},\boldsymbol{w})}{\partial \boldsymbol{c}(\boldsymbol{x},\boldsymbol{w})} \tag{5.20}$$

where the derivatives of transfer function response H with respect to the pole/residues (i.e., $\frac{\partial H(s,\boldsymbol{x},\boldsymbol{w})}{\partial \boldsymbol{c}(\boldsymbol{x},\boldsymbol{w})}$) are calculated similarly using (5.8)-(5.11). The derivatives of the pole/residues with respect to the neural network weights $\boldsymbol{w}$ are defined by neural network derivative formulation in the reference [14].

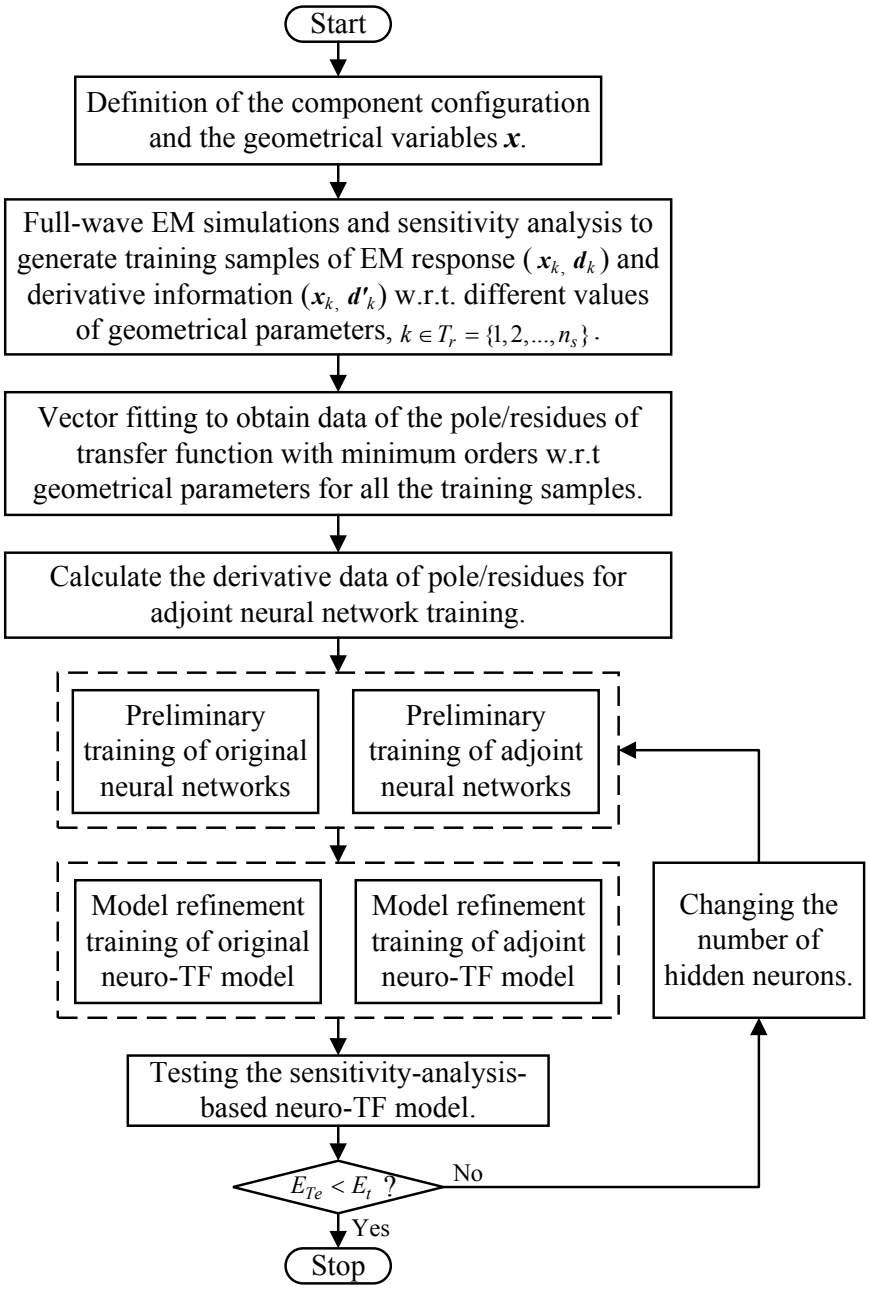


Figure 5.3: The flowchart of the overall development process for the sensitivity-analysis-based neuro-TF model.

The gradient-based training is also performed simultaneously in the adjoint neuro-TF model. The training requires the derivatives of the adjoint neuro-TF model output $d\boldsymbol{y}/d\boldsymbol{x}$ w.r.t the weighting parameters $\boldsymbol{w}$, i.e., the second-order derivatives of the model outputs $\boldsymbol{y}$ (real/imaginary parts of the transfer function response H) w.r.t $\boldsymbol{x}$ and $\boldsymbol{w}$, formulated as,

$$\frac{\partial^2 H(s,\boldsymbol{x},\boldsymbol{w})}{\partial \boldsymbol{w}\partial \boldsymbol{x}^T} = \frac{\partial \boldsymbol{c}^T}{\partial \boldsymbol{w}} \frac{\partial^2 H}{\partial \boldsymbol{c}\partial \boldsymbol{c}^T} \frac{\partial \boldsymbol{c}}{\partial \boldsymbol{x}^T} + \sum_{i=1}^{4N} \frac{\partial H}{\partial c_i} \frac{\partial^2 c_i}{\partial \boldsymbol{w}\partial \boldsymbol{x}^T} \tag{5.21}$$

where c_i represents the i^{th} element in the $\boldsymbol{c}$ vector.

To obtain this second-order derivative, we need to first calculate the matrix of second-order derivatives of transfer function response H with respect to the vector $\boldsymbol{c}$ which contains real and imaginary parts of poles and residues (i.e., $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\eta}$ and $\boldsymbol{\xi}$). The elements in this matrix are calculated as

$$\frac{\partial^2 H}{\partial \alpha_i \partial \alpha_t} = \begin{cases} \dfrac{2r_i}{(s-p_i)^3} + \dfrac{2r_i^*}{(s-p_i^*)^3} = \phi_i^{(1)}, & i = t \\ 0, & i \neq t \end{cases} \tag{5.22}$$

$$\frac{\partial^2 H}{\partial \alpha_i \partial \beta_t} = \begin{cases} \dfrac{j2r_i}{(s-p_i)^3} - \dfrac{j2r_i^*}{(s-p_i^*)^3} = \phi_i^{(2)}, & i = t \\ 0, & i \neq t \end{cases} \tag{5.23}$$

$$\frac{\partial^2 H}{\partial \alpha_i \partial \eta_t} = \begin{cases} \dfrac{1}{(s-p_i)^2} + \dfrac{1}{(s-p_i^*)^2} = \phi_i^{(3)}, & i = t \\ 0, & i \neq t \end{cases} \tag{5.24}$$

$$\frac{\partial^2 H}{\partial \alpha_i \partial \xi_t} = \begin{cases} \dfrac{j}{(s-p_i)^2} - \dfrac{j}{(s-p_i^*)^2} = \phi_i^{(4)}, & i = t \\ 0, & i \neq t \end{cases} \tag{5.25}$$

$$\frac{\partial^2 H}{\partial \beta_i \partial \beta_t} = -\frac{\partial^2 H}{\partial \alpha_i \partial \alpha_t} \tag{5.26}$$

$$\frac{\partial^2 H}{\partial \beta_i \partial \eta_t} = \frac{\partial^2 H}{\partial \alpha_i \partial \xi_t} \tag{5.27}$$

$$\frac{\partial^2 H}{\partial \beta_i \partial \xi_t} = -\frac{\partial^2 H}{\partial \alpha_i \partial \eta_t} \tag{5.28}$$

$$\frac{\partial^2 H}{\partial \eta_i \partial \eta_t} = \frac{\partial^2 H}{\partial \eta_i \partial \xi_t} = \frac{\partial^2 H}{\partial \xi_i \partial \xi_t} = 0 \tag{5.29}$$

where $\phi_i^{(1)}$, $\phi_i^{(2)}$, $\phi_i^{(3)}$, $\phi_i^{(4)}$ are used to simplify the representation of the elements in (5.22)-(5.25).

The second-order derivative of the transfer function response H w.r.t model

inputs $\boldsymbol{x}$ and neural network weights $\boldsymbol{w}$ is calculated using a simplified (5.30).

$$\begin{aligned}\frac{\partial^2 H}{\partial \boldsymbol{w} \partial \boldsymbol{x}^T} = &\sum_{i=1}^{N} \left(\left[\frac{\partial \alpha_i}{\partial \boldsymbol{w}} \; \frac{\partial \beta_i}{\partial \boldsymbol{w}} \; \frac{\partial \eta_i}{\partial \boldsymbol{w}} \; \frac{\partial \xi_i}{\partial \boldsymbol{w}} \right] \begin{bmatrix} \phi_i^{(1)} & \phi_i^{(2)} & \phi_i^{(3)} & \phi_i^{(4)} \\ \phi_i^{(2)} & -\phi_i^{(1)} & \phi_i^{(4)} & -\phi_i^{(3)} \\ \phi_i^{(3)} & \phi_i^{(4)} & 0 & 0 \\ \phi_i^{(4)} & -\phi_i^{(3)} & 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{\partial \alpha_i}{\partial \boldsymbol{x}^T} \\ \frac{\partial \beta_i}{\partial \boldsymbol{x}^T} \\ \frac{\partial \eta_i}{\partial \boldsymbol{x}^T} \\ \frac{\partial \xi_i}{\partial \boldsymbol{x}^T} \end{bmatrix} \right) \\ &+ \sum_{i=1}^{N} \left(\left[\frac{\partial H}{\partial \alpha_i} \; \frac{\partial H}{\partial \beta_i} \; \frac{\partial H}{\partial \eta_i} \; \frac{\partial H}{\partial \xi_i} \right] \begin{bmatrix} \frac{\partial^2 \alpha_i}{\partial \boldsymbol{w} \partial \boldsymbol{x}^T} \\ \frac{\partial^2 \beta_i}{\partial \boldsymbol{w} \partial \boldsymbol{x}^T} \\ \frac{\partial^2 \eta_i}{\partial \boldsymbol{w} \partial \boldsymbol{x}^T} \\ \frac{\partial^2 \xi_i}{\partial \boldsymbol{w} \partial \boldsymbol{x}^T} \end{bmatrix} \right)\end{aligned} \tag{5.30}$$

This second order derivative contains the calculation of the first and second order derivatives of the transfer function response with respect to pole/residues, and the first and second order derivatives of pole/residues with respect to the geometrical parameters $\boldsymbol{x}$ and neural network internal weights $\boldsymbol{w}$. The first order derivatives of the transfer function response with respect to pole/residues are calculated using (5.8)-(5.11). The second order derivative matrix of transfer function response with respect to pole/residues are simplified and calculated using (5.22)-(5.29). The first and second order derivatives of pole/residues with respect to the geometrical parameters $\boldsymbol{x}$ and neural network internal weights $\boldsymbol{w}$ are calculated similarly to [14]. The overall second order derivatives of H w.r.t $\boldsymbol{x}$ and $\boldsymbol{w}$ are calculated for use in the training of the first order derivatives of the neuro-TF model (i.e., training of the adjoint neuro-TF model).

The calculation of the derivatives of $\boldsymbol{y}$ and $d\boldsymbol{y}/d\boldsymbol{x}$ with respect to $\boldsymbol{w}$ is used for

the gradient-based training process of the proposed sensivity-analysis-based neuro-TF model. Using these derivative calculations, we can obtain the gradient information of the model faster and more accurately than using brute-force perturbation method, thus speeding up the training process.

5.3 Application Examples

5.3.1 Parametric Modeling of a Microwave Junction

The development of proposed sensitivity-analysis-based neuro-TF model for parametric modeling of the EM response of a microwave junction [14] is illustrated in this example, as shown in Fig. 5.4. g represents the gap between the edges of the two conductive walls in the junction cavity. w_s represents the width of the slot on the upper conductive wall. l_s represents the length of the slot on the upper conductive wall. h_c represents the height of the cylinder in the junction cavity. r_c represents the radius of the cylinder in the junction cavity. t_w represents thickness of the conductive walls. h represents the height of the junction cavity.

Fig. 5.5 shows the structure of the proposed sensitivity-analysis-based neuro-TF model for the junction example. This model has eight input variables including seven geometrical parameters and frequency, i.e., $[g\ h_c\ r_c\ t_w\ w_s\ l_s\ h\ \omega]^T$. The model has 64 outputs, i.e.,

$$[RS_{11}\ IS_{11}\ RS_{21}\ IS_{21}\ RS_{31}\ IS_{31}\ RS_{41}\ IS_{41}\ \frac{dRS_{11}}{dg}$$
$$\frac{dRS_{11}}{dh_c}\ \frac{dRS_{11}}{dr_c}\ \frac{dRS_{11}}{dt_w}\ \frac{dRS_{11}}{dw_s}\ \frac{dRS_{11}}{dl_s}\ \cdots\ \frac{dIS_{41}}{dh}]^T,$$

The model outputs include the real/imaginary part of S_{11}, S_{21}, S_{31} and S_{41}, and their

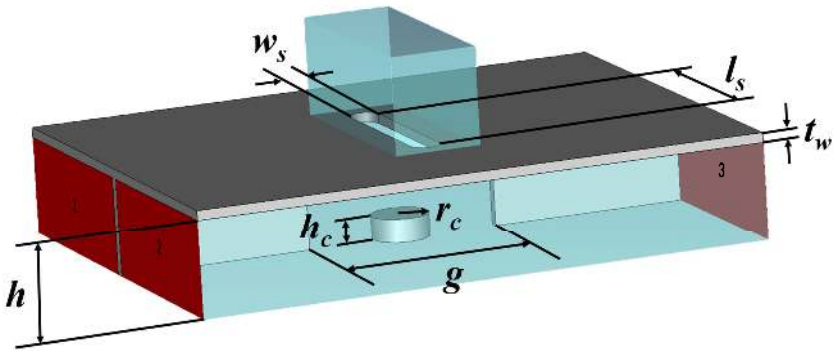


Figure 5.4: The geometrical parameters of the microwave junction example and the 3D configuration for EM simulation. The development of a parametric model with these seven design variables will be performed.

derivatives w.r.t seven input geometrical variables. The full-wave EM simulation and EM sensitivity analysis for generating training and testing data is performed using the *CST Studio Suite* 2014 software. The geometrical parameters g, h_c and r_c are considered as sensitivity variables in EM sensitivity analysis. The derivatives of real and imaginary parts of S_{11}, S_{21}, S_{31}, and S_{41} with respect to three sensitivity variables g, h_c and r_c are obtained from the EM sensitivity analysis. The other geometrical variables (i.e., t_w, w_s, l_s, and h) are considered as non-sensitivity variables because those variables are not available for EM sensitivity analysis. We set the training weights for the corresponding outputs of the adjoint neuro-TF model, i.e., weights $b_{j,i}$ for

$$\left[\frac{dRS_{11}}{dt_w}\ \frac{dRS_{11}}{dw_s}\ \frac{dRS_{11}}{dl_s}\ \frac{dRS_{11}}{dh}\right.$$
$$\left.\frac{dIS_{11}}{dt_w}\ \frac{dIS_{11}}{dw_s}\ \frac{dIS_{11}}{dl_s}\ \cdots\ \frac{dIS_{41}}{dh}\right]^T,$$

to be zero during the training process. For the other outputs of the adjoint neuro-

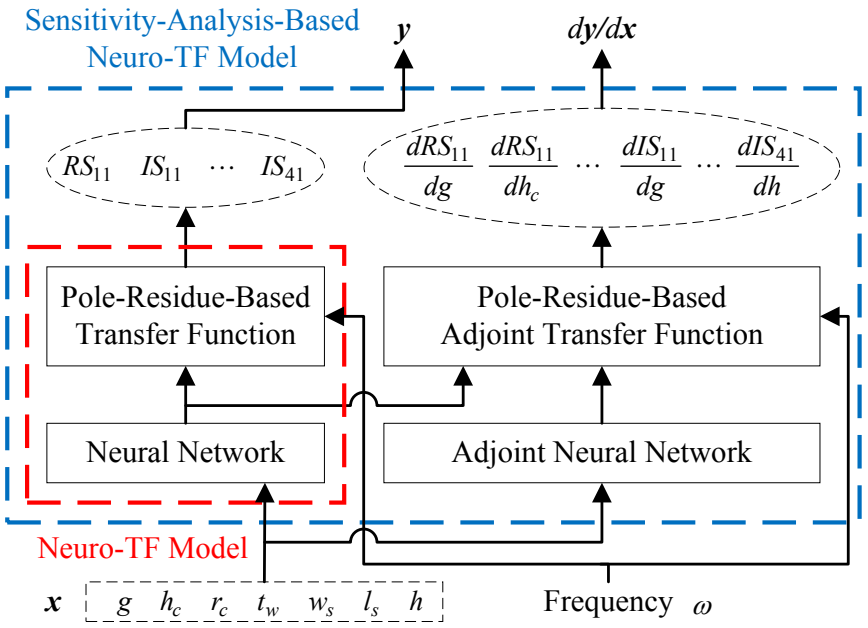


Figure 5.5: The structure of the proposed sensitivity-analysis-based neuro-TF model for the junction example. Seven geometrical parameters $[g\ h_c\ r_c\ t_w\ w_s\ l_s\ h]$ are used as the input variables of the junction model. The outputs of the original neuro-TF model $\boldsymbol{y}$ are the S-parameters, while the outputs of the adjoint neuro-TF model $d\boldsymbol{y}/d\boldsymbol{x}$ are the derivatives of S-parameters with respect to the geometrical parameters.

TF model which represent the derivatives of the outputs of the original neuro-TF model with respect to the sensitivity variables, the weights $b_{j,i}$ are set to be non-zero values during the training process.

Two different cases defined in Table 5.1 are used to illustrate the proposed method. Case 1 is with a narrower parameter range and Case 2 is with a wider parameter range. Design of experiments (DOE) [122] method is used as the sampling method for both training and testing data. As shown in Table 5.1, 7 levels of DOE are used for defining samples of the training data, i.e., a total of 49 samples of

Table 5.1: Definition of Training and Testing Data for the Junction Example

	Geometrical Parameters		Training Data (49 samples)			Testing Data (144 samples)		
			Min	Max	Step	Min	Max	Step
Case 1	Sensitivity Variables	g (mm)	17	23	1	17.25	22.75	0.5
		h_c (mm)	1.8	3.0	0.2	1.85	3.05	0.1
		r_c (mm)	2.4	3.6	0.2	2.35	3.55	0.1
	t_w (mm)		0.88	1.12	0.04	0.89	1.11	0.02
	w_s (mm)		3	3.48	0.08	3.02	3.46	0.04
	l_s (mm)		20	20.48	0.08	20.02	20.46	0.04
	h (mm)		10	10.48	0.08	10.02	10.46	0.04
Case 2	Sensitivity Variables	g (mm)	14	26	2	14.5	25.5	1
		h_c (mm)	1.2	3.6	0.4	1.3	3.5	0.2
		r_c (mm)	1.8	4.2	0.4	1.9	4.1	0.2
	t_w (mm)		0.88	1.12	0.04	0.89	1.11	0.02
	w_s (mm)		2.4	3.6	0.2	2.45	3.55	0.1
	l_s (mm)		19.4	20.6	0.2	19.45	20.55	0.1
	h (mm)		10	11.2	0.2	10.05	11.15	0.1

training data; while 12 levels of DOE are used for the testing data, i.e., a total of 144 samples of testing data. All the simulations are done on the computers with *Intel Xeon X5680* processor. The frequency range for this junction example is 7 GHz-9 GHz. The *NeuroModelerPlus* software is used to train the proposed sensitivity-analysis-based neuro-TF model. Although the overall model for training involved the original neuro-TF model and adjoint neuro-TF model, the final parametric model is simple, i.e., the original neuro-TF model.

In both cases, the orders N_k of transfer functions vary from 6 to 8 among different samples of geometrical parameters. The comparisons of training/testing errors in two cases using proposed pole-residue tracking technique with sensitivity infor-

mation and existing pole-residue tracking technique without sensitivity information [30] are illustrated in Table 5.2. In Case 1, since the geometrical step sizes between the training samples are relatively small, both methods obtain the correct pole splitting, resulting in the data of the pole/residues to be the same among both techniques for preliminary training of the model. Thus the training and testing errors are both good using both methods. In Case 2, when the geometrical step sizes between the training samples are large, part of the pole splitting is incorrect using the existing pole-residue tracking technique without sensitivity information [30]. The incorrect pole splitting causes abrupt changes in the data of poles and residues. That results in a relatively large training and testing error in preliminary training of the neural networks, thus leading to a relatively large error in the refinement training of the overall model. By using the proposed pole-residue tracking technique with the sensitivity information, we can obtain the correct pole splitting resulting in the smooth changes of poles and residues with respect to the changes in geometrical parameters even when the geometrical step sizes are large. Thus a better modeling accuracy is obtained by using the proposed pole-residue tracking technique.

For comparison purpose, we also perform the modeling using existing SAANN technique [14]. This technique uses the data of EM simulation and derivatives to train a pure neural network (i.e., without transfer function). Table 5.3 illustrates two cases of comparisons of training and testing errors among the SAANN Model [14] using fewer data (49 training samples), existing neuro-TF model [30] using fewer data (49 training samples), existing neuro-TF model using more data (169

Table 5.2: Comparisons of Different Pole-Residue Tracking Techniques for the Junction Example

	Pole-Residue Tracking Technique	Number of Hidden Neurons	ANN Training Error	ANN Testing Error	Overall Model Training Error	Overall Model Testing Error
Case 1	Existing Technique without Sensitivity	8	1.923%	2.124%	0.577%	0.612%
	Proposed Technique with Sensitivity	8	1.923%	2.124%	0.577%	0.612%
Case 2	Existing Technique without Sensitivity	10	2.956%	3.245%	1.116%	1.259%
	Existing Technique without Sensitivity	12	2.174%	4.586%	0.658%	1.785%
	Proposed Technique with Sensitivity	10	2.186%	2.265%	0.670%	0.832%

training samples), and the proposed model using fewer data (49 training samples). In Case 1, all methods obtain relatively small training and testing errors, because the geometrical parameters vary within a small range. In Case 2, when the range of geometrical parameters gets wider, the SAANN model [14] and existing neuro-TF model trained with fewer training data cannot obtain good testing accuracy even though they both maintain similarly good training accuracy. For the SAANN model, many hidden neurons are needed to represent the frequency response for each sample. The proposed model uses much less hidden neurons because the frequency responses are represented by the transfer function. Thus the nonlinearity of the SAANN model is much higher than the proposed model. This high nonlinearity leads to the testing errors of SAANN model to be higher than the proposed model

when the range of geometrical parameters gets wider. For the existing neuro-TF model, with large number of training data, the model can be trained to be very accurate. When the number of data for training becomes less in the same geometrical range, the model will be less accurate. The proposed model is trained with not only the EM simulation data but also the derivative information, thus the model accuracy of the proposed model is much better than that of the existing neuro-TF model when both model are trained with few data samples. Fig. 5.6 shows the comparisons of the derivatives of the real parts of S_{11} with respect to the sensitivity variables g, h_c and r_c in existing neuro-TF model using fewer and more data, proposed model using fewer data and CST sensitivity analysis for one training sample $\boldsymbol{x} = [18\ 3.2\ 3.4\ 0.84\ 2.8\ 19.8\ 11.2]^T$ (mm). As shown in Fig. 5.6, the proposed model has more accurate derivative information after training process, which helps to obtain more accurate testing results.

The comparisons of the outputs of the proposed model, existing neuro-TF model developed using different size of data and EM data for three different testing geometrical parameters #1, #2 and #3 are shown in Fig. 5.7. The values of input geometrical variables for three different testing samples of junctions are shown as follows.

Test geometrical sample

#1 : $\boldsymbol{x} = [25.5\ 2.7\ 3.1\ 0.97\ 2.95\ 19.95\ 10.75]^T$ (mm)

Test geometrical sample

#2 : $\boldsymbol{x} = [22.5\ 3.1\ 2.5\ 0.91\ 2.75\ 19.95\ 10.95]^T$ (mm)

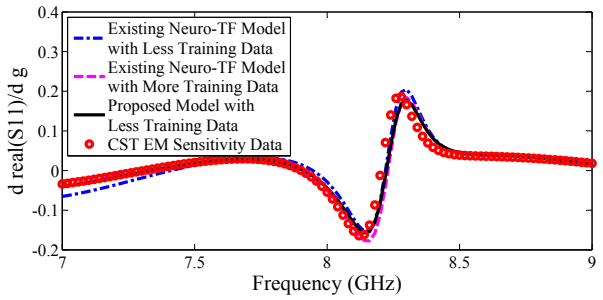
Test geometrical sample

Table 5.3: Comparisons of Different Modeling Methods for the Junction Example

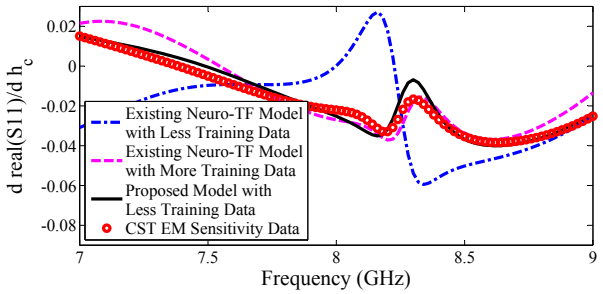
	Model Methods	Number of Hidden Neurons	Average Training Error	Average Testing Error
Case 1	SAANN Model Using 49 Samples of Training Data	15	0.592%	0.632%
	Existing Neuro-TF Model Using 49 Samples of Training Data	8	0.538%	0.673%
	Proposed Model Using 49 Samples of Training Data	8	0.577%	0.612%
Case 2	SAANN Model Using 49 Samples of Training Data	20	0.792%	2.264%
	Existing Neuro-TF Model Using 49 Samples of Training Data	8	0.771%	8.035%
	Existing Neuro-TF Model Using 169 Samples of Training Data	15	0.859%	0.962%
	Proposed Model Using 49 Samples of Training Data	10	0.670%	0.832%

$\#3: \ \boldsymbol{x} = [15.5 \ 2.1 \ 3.5 \ 1.01 \ 2.85 \ 19.85 \ 10.35]^T$ (mm)

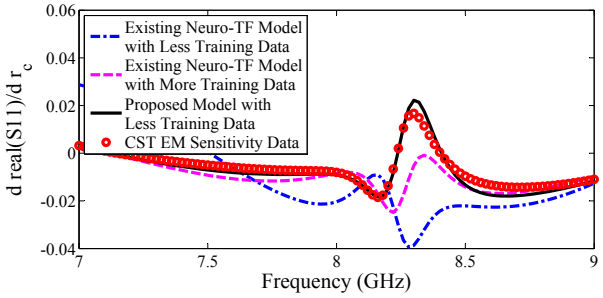
The three testing samples are not used during training process. The development cost of the proposed neuro-TF model for this junction example is around 0.82 h, which consists of data generation cost (49 training samples) and training cost. For the development of existing neuro-TF model (169 training samples), the cost is around 2.36 h. As shown in Table 5.3 and Fig. 5.7, to achieve good model accuracy, fewer training data is needed for the proposed model than that needed for the existing neuro-TF model.



(a)

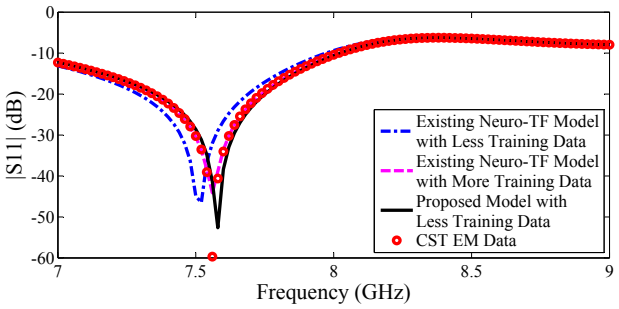


(b)

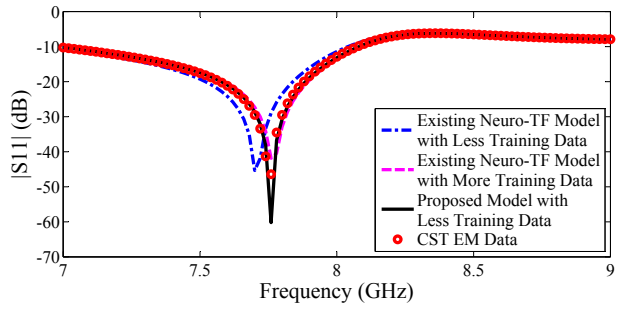


(c)

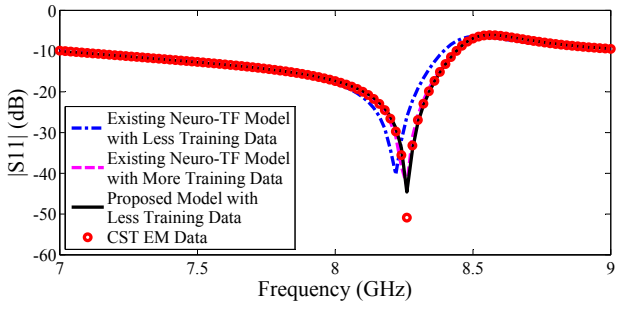
Figure 5.6: Comparison of the derivatives of the existing neuro-TF model using fewer and more data, the proposed model using fewer data and CST EM sensitivity data for this junction example: (a) derivatives of real parts of S_{11} with respect to sensitivity variable g, (b) derivatives of real parts of S_{11} with respect to sensitivity variable h_c, and (c) derivatives of real parts of S_{11} with respect to sensitivity variable r_c. The derivatives of the proposed sensitivity-analysis-based neuro-TF model matches with EM sensitivity data from CST simulator much more closely than the existing method as shown in the figure.



(a)



(b)



(c)

Figure 5.7: Comparisons of S_{11} of the existing neuro-TF model using fewer and more data, the proposed model using fewer data and CST EM data for the junction example: (a) test geometrical sample #1, (b) test geometrical sample #2, and (c) test geometrical sample #3. With the same size of data, the proposed model can obtain a better accuracy than the existing neuro-TF model as shown in the figure. To achieve good model accuracy, fewer training data is needed for the proposed model than that needed for the existing neuro-TF model.

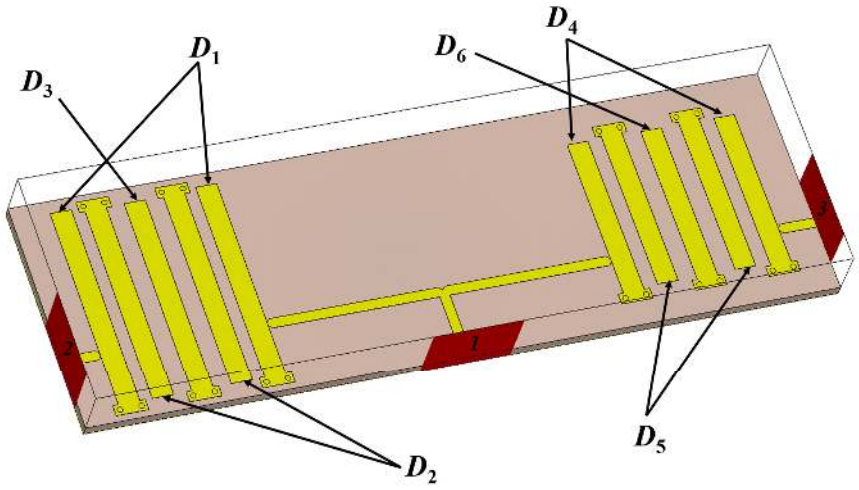


Figure 5.8: The geometrical parameters of the diplexer example and the 3D configuration for EM simulation. The development of a parametric model with these six design variables will be performed.

5.3.2 Parametric Modeling of a Diplexer

This example illustrates the development of pole-residue-based neuro-TF model for parametric modeling of the EM behavior of a diplexer [14], shown in Fig. 5.8. D_1, D_2, D_3, D_4, D_5, and D_6 are the offset distances from the end of each microstrip line to its corresponding fringe.

Fig. 5.9 shows the structure of the proposed sensitivity-analysis-based neuro-TF model for the diplexer example. This model has seven input variables including six geometrical parameters and frequency, i.e., $[D_1\ D_2\ D_3\ D_4\ D_5\ D_6\ \omega]^T$. The model has 42 outputs, i.e.,

$$[RS_{11}\ IS_{11}\ RS_{21}\ IS_{21}\ RS_{31}\ IS_{31}\ \frac{dRS_{11}}{dD_1}\ \frac{dRS_{11}}{dD_2}\ \frac{dRS_{11}}{dD_3}\ \frac{dRS_{11}}{dD_4}\ \frac{dRS_{11}}{dD_5}\ \frac{dRS_{11}}{dD_6}\ \cdots\ \frac{dIS_{31}}{dD_6}]^T,$$

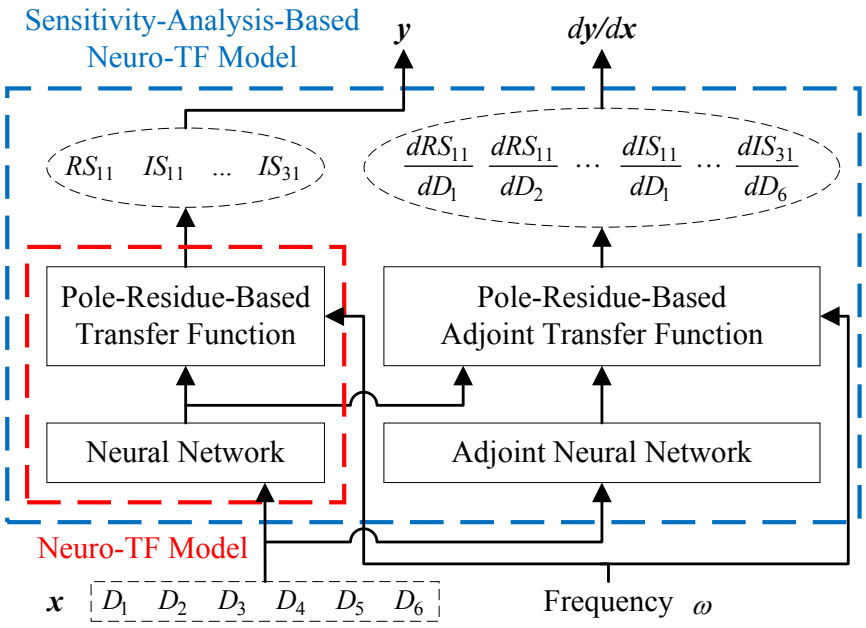


Figure 5.9: The structure of the proposed sensitivity-analysis-based neuro-TF model for the diplexer example. Six geometrical parameters $[D_1\ D_2\ D_3\ D_4\ D_5\ D_6]$ are used as the input variables of the diplexer model. The outputs of the original neuro-TF model $\boldsymbol{y}$ are the S-parameters, while the outputs of the adjoint neuro-TF model $d\boldsymbol{y}/d\boldsymbol{x}$ are the derivatives of S-parameters with respect to the geometrical parameters.

The model outputs include the real/imaginary part of S_{11}, S_{21}, and S_{31}, and their derivatives w.r.t six input geometrical variables. Training and testing data are generated using the *CST Studio Suite* 2014 software. Every geometrical variable in this example is considered as sensitivity variable. The sensitivity analysis is performed by using EM simulator to generate the derivative data of real/imaginary part of S_{11}, S_{21}, and S_{31} with respect to six sensitivity variables D_1, D_2, D_3, D_4, D_5, and D_6.

Two different cases defined in Table 5.4 are used to illustrate the proposed

Table 5.4: Definition of Training and Testing Data for the Diplexer Example

Geometrical Parameters			Training Data (25 samples)			Testing Data (64 samples)		
			Min	Max	Step	Min	Max	Step
Case 1	Sensitivity Variables	D_1 (mm)	2	10	2	2.5	9.5	1
		D_2 (mm)	-6	2	2	-5.5	1.5	1
		D_3 (mm)	-6	2	2	-5.5	1.5	1
		D_4 (mm)	2	10	2	2.5	9.5	1
		D_5 (mm)	-6	2	2	-5.5	1.5	1
		D_6 (mm)	-6	2	2	-5.5	1.5	1
Case 2	Sensitivity Variables	D_1 (mm)	-4	12	4	-3	11	2
		D_2 (mm)	-12	4	4	-11	3	2
		D_3 (mm)	-12	4	4	-11	3	2
		D_4 (mm)	-4	12	4	-3	11	2
		D_5 (mm)	-12	4	4	-11	3	2
		D_6 (mm)	-12	4	4	-11	3	2

method. Case 1 is with a narrower parameter range and Case 2 is with a wider parameter range. Table 5.4 shows the definition of training and testing data. DOE sampling method is used for both training and testing data. As shown in Table 5.4, 5 levels of DOE are used for defining samples of the training data, i.e., a total of 25 samples of training data; while 8 levels of DOE are used for the testing data, i.e., a total of 64 samples of testing data. The frequency range for this diplexer example is 1.5 GHz-3.5 GHz. The *NeuroModelerPlus* software is used for developing the proposed sensitivity-analysis-based neuro-TF model. After training, the final model used for design optimization only consists the original neuro-TF model.

In both cases, the orders N_k of transfer functions vary from 16 to 20 among different samples of geometrical parameters. Table 5.5 illustrates the comparison

of training and testing errors in two cases using proposed pole-residue tracking technique with sensitivity information and existing pole-residue tracking technique without sensitivity information [30]. In Case 1, since the geometrical step sizes between the training samples are relatively small, both methods obtain the correct pole splitting, leading to good training and testing results. In Case 2, when the geometrical step sizes between the training samples are large, part of the pole splitting is incorrect using the existing pole-residue tracking technique without sensitivity information [30]. This results in a relatively large error in both the preliminary training of the neural networks and the refinement training of the overall model. By using the proposed pole-residue tracking technique with the sensitivity information, we can obtain a better modeling accuracy even when the geometrical step sizes are large.

For comparison purpose, we also perform the modeling using existing SAANN technique [14]. Table 5.6 illustrates two cases of comparisons of training and testing errors among the SAANN Model using fewer data (25 training samples), existing neuro-TF model [30] using fewer data (25 training samples), existing neuro-TF model using more data (64 training samples), and the proposed model using fewer data (25 training samples). In Case 1, all methods obtain relatively small training and testing errors, because the geometrical parameters vary within a small range. In Case 2, when the range of geometrical parameters gets wider, the SAANN model and existing neuro-TF model trained with fewer training data cannot obtain good testing accuracy even though they both maintain similarly good training accuracy. Fig. 5.10 shows the comparisons of the derivatives of the real part of S_{11} with

Table 5.5: Comparisons of Different Pole-Residue Tracking Techniques for the Diplexer Example

	Pole-Residue Tracking Technique	Number of Hidden Neurons	ANN Training Error	ANN Testing Error	Overall Model Training Error	Overall Model Testing Error
Case 1	Original Technique without Sensitivity	12	4.126%	4.425%	0.892%	0.915%
	Proposed Technique with Sensitivity	12	4.126%	4.425%	0.892%	0.915%
Case 2	Original Technique without Sensitivity	12	6.246%	6.235%	1.786%	1.913%
	Original Technique without Sensitivity	15	4.358%	8.325%	1.143%	5.126%
	Proposed Technique with Sensitivity	12	4.425%	4.532%	1.065%	1.254%

respect to the sensitivity variables D_1, D_2 and D_3 in existing neuro-TF model using fewer and more data, proposed model using fewer data and CST sensitivity analysis for one training sample $\boldsymbol{x} = [2\ -6\ -6\ 2\ -6\ -6]^T$ (mm). As shown in Fig. 5.10, the proposed model has more accurate derivative information after training process, which helps to obtain more accurate testing results.

The comparisons of the outputs of the proposed model, existing neuro-TF model developed using different size of training data and EM data for three different testing geometrical parameters #1, #2 and #3 are shown in Fig. 5.11. The values of input geometrical variables for three different testing samples of diplexers are shown as follows.

Test geometrical sample

Table 5.6: Comparisons of Different Modeling Methods for the Diplexer Example

	Model Type	Number of Hidden Neurons	Average Training Error	Average Testing Error
Case 1	SAANN Model Using 25 Samples of Training Data	30	0.906%	0.924%
	Original Neuro-TF Model Using 25 Samples of Training Data	10	0.867%	0.932%
	Proposed Model Using 25 Samples of Training Data	12	0.892%	0.915%
Case 2	SAANN Model Using 25 Samples of Training Data	36	1.165%	11.564%
	Original Neuro-TF Model Using 25 Samples of Training Data	12	1.095%	9.264%
	Original Neuro-TF Model Using 81 Samples of Training Data	18	1.135%	1.368%
	Proposed Model Using 25 Samples of Training Data	12	1.065%	1.254%

$\#1: \ \boldsymbol{x} = [-3 \ -9 \ -9 \ 1 \ -5 \ -3]^T$ (mm)

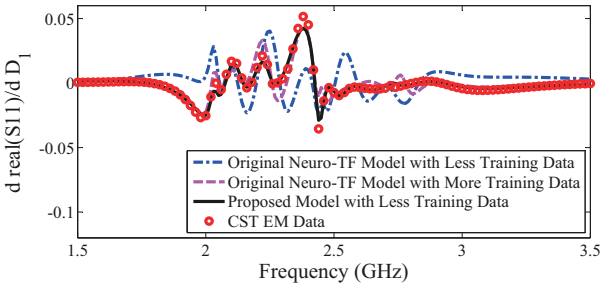
Test geometrical sample

$\#2: \ \boldsymbol{x} = [-1 \ -1 \ 1 \ 7 \ -5 \ -11]^T$ (mm)

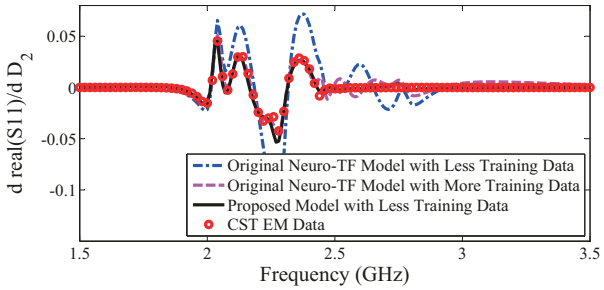
Test geometrical sample

$\#3: \ \boldsymbol{x} = [11 \ -1 \ -3 \ 1 \ -11 \ -5]^T$ (mm)

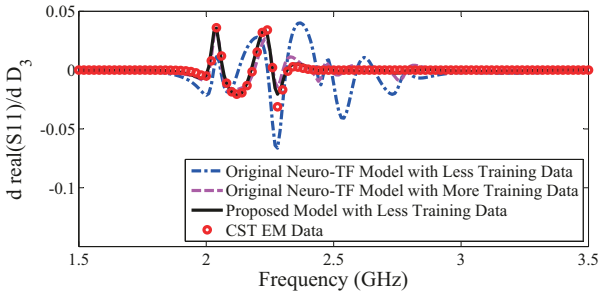
The magnitude of S_{21} and S_{31} of the proposed model and their comparisons with EM data for the testing geometrical parameters #1 are shown in Fig. 5.12. From Fig. 5.11 and Fig. 5.12, because the proposed model is trained very accurately (error around 1%) using real and imaginary parts of S-parameters, the dB values of S-parameters for the proposed model and the data can match well when the values



(a)

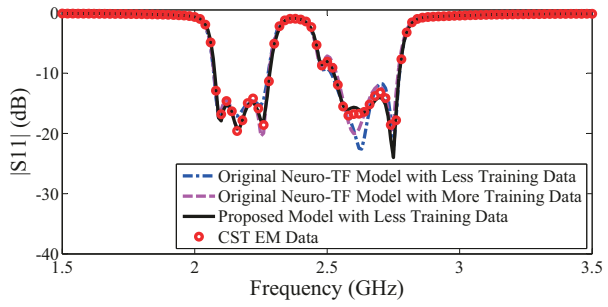


(b)

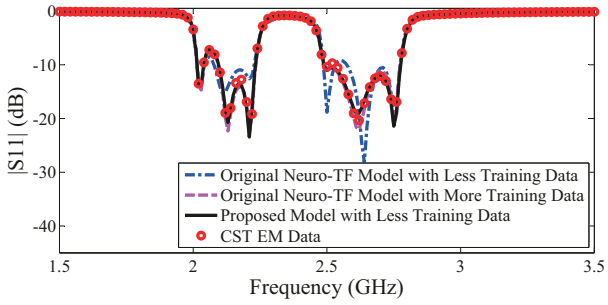


(c)

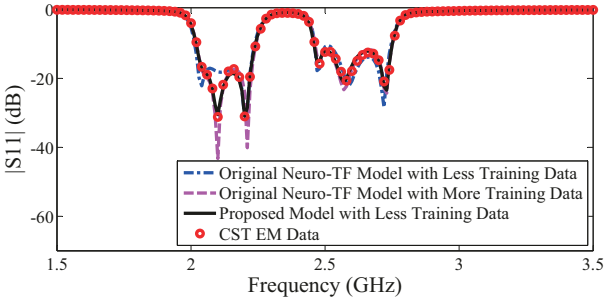
Figure 5.10: Comparison of the derivatives of the existing neuro-TF model using fewer and more data, the proposed model using fewer data and CST EM sensitivity data for this diplexer example: (a) derivatives of real parts of S_{11} with respect to sensitivity variable D_1, (b) derivatives of real parts of S_{11} with respect to sensitivity variable D_2, and (c) derivatives of real parts of S_{11} with respect to sensitivity variable D_3. The derivatives of the proposed sensitivity-analysis-based neuro-TF model matches with EM sensitivity data from CST simulator much more closely than the existing method as shown in the figure.



(a)



(b)



(c)

Figure 5.11: Comparisons of S_{11} of the existing neuro-TF model using fewer data and more data, the proposed model using fewer data and CST EM data: (a) test geometrical sample #1, (b) test geometrical sample #2, and (c) test geometrical sample #3 for the diplexer example. With the same size of data, the proposed model can obtain a better accuracy than the existing neuro-TF model as shown in the figure. To achieve good model accuracy, fewer training data is needed for the proposed model than that needed for the existing neuro-TF model.

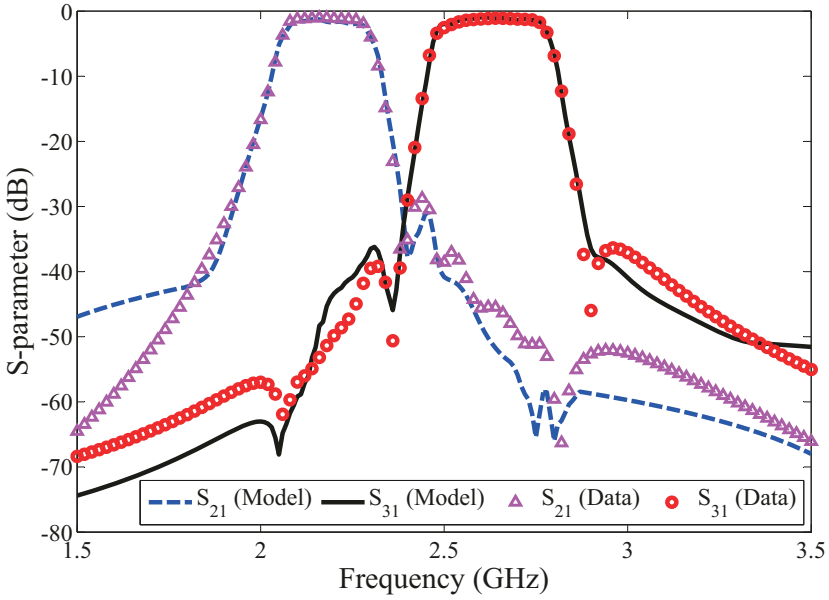


Figure 5.12: Comparisons of S_{21} and S_{31} (magnitude in decibels) of the trained proposed neuro-TF model and EM data of test geometrical sample #1 for the diplexer example.

of S-parameters are above -20 dB. However when the test data are below -20dB, the dB values of S-parameters for the proposed model begin to differ from the data as shown in Fig. 5.11 and Fig. 5.12.

The three testing samples are not used during training process. The development cost of the proposed neuro-TF model for this diplexer example is around 3.56 h, which consists of data generation cost (25 training samples) and training cost. For the development of existing neuro-TF model (81 geometrical samples of training data), the cost is around 7.69 h. As shown in Table 5.6 and Fig. 5.11, to achieve good model accuracy, fewer training data is needed for the proposed model than that needed for the existing neuro-TF model.

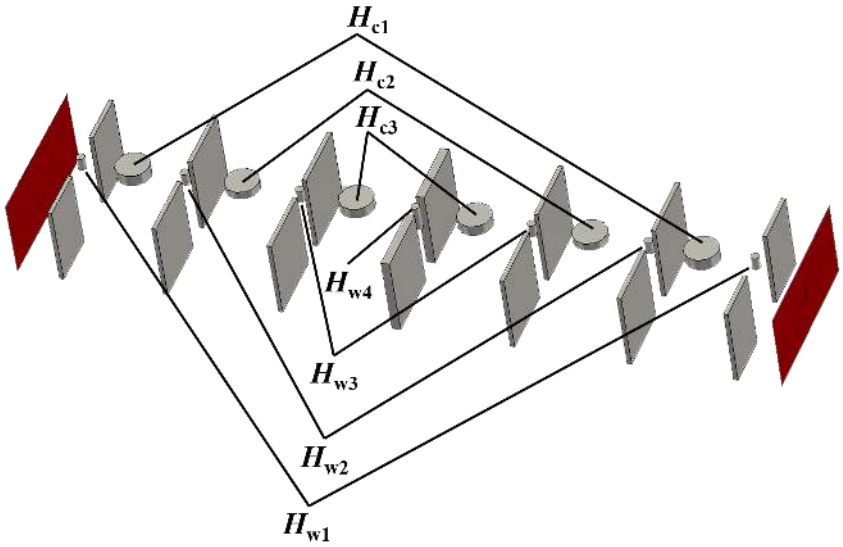


Figure 5.13: The geometrical parameters of the microwave cavity filter example and the 3D configuration for EM simulation. The waveguide enclosure of this structure is eliminated in the figure in order to indicate the geometrical parameters more clearly. The development of a parametric model with these seven design variables will be developed.

5.3.3 Parametric Modeling of a Microwave Cavity Filter

In this example, we illustrate the parametric modeling using the proposed pole-residue-based neuro-TF model for the EM response of a microwave cavity filter [14], shown in Fig. 5.13, where H_{c1}, H_{c2}, and H_{c3} are the heights of the large cylinders positioned at the cavity centers; H_{w1}, H_{w2}, H_{w3}, and H_{w4} are the heights of the small cylinders between the walls.

Fig. 5.14 shows the structure of proposed model for the microwave cavity filter example. This model has eight input variables including seven geometrical parameters and frequency, i.e., $[H_{c1}\ H_{c2}\ H_{c3}\ H_{w1}\ H_{w2}\ H_{w3}\ H_{w4}\ \omega]^T$. The model has 16

outputs, i.e.,

$$[RS_{11}\ IS_{11}\ \frac{dRS_{11}}{dH_{c1}}\ \frac{dRS_{11}}{dH_{c2}}\ \frac{dRS_{11}}{dH_{c3}}\ \frac{dRS_{11}}{dH_{w1}}\ \frac{dRS_{11}}{dH_{w2}}\ \frac{dRS_{11}}{dH_{w3}}\ \cdots\ \frac{dIS_{11}}{dH_{w4}}]^T,$$

The model outputs include the real/imaginary part of S_{11}, and their derivatives w.r.t seven input geometrical variables. Training and testing data are generated using the *CST Studio Suite* 2014 software. Every geometrical variable in this example is considered as sensitivity variable. The sensitivity analysis is performed by using EM simulator to generate the derivative data of real/imaginary part of S_{11} with respect to seven sensitivity variables H_{c1}, H_{c2}, H_{c3}, H_{w1}, H_{w2}, H_{w3}, and H_{w4}.

Two different cases defined in Table 5.7 are used to illustrate the proposed method. Case 1 is with a narrower parameter range and Case 2 is with a wider parameter range. Table 5.7 shows the ranges of training and testing data. The average sensitivities of S-parameters with respect to H_{c1}, H_{c2}, and H_{c3} obtained by EM sensitivity analysis are around ten times larger than those for H_{w1}, H_{w2}, H_{w3}, and H_{w4}. Thus, the ranges for H_{c1}, H_{c2}, and H_{c3} are determined to be ten times smaller than those for H_{w1}, H_{w2}, H_{w3}, and H_{w4}. Design of experiments (DOE) method is used as the sampling method for both training and testing data. DOE sampling method is an orthogonal sampling method which is used for generating multiple sample points. Orthogonal sampling around the central point enables the surrogate model to use far fewer sampling points compared to full-grid distribution. As shown in Table 5.7, 7 levels of DOE are used for defining samples of the training data, i.e., a total of 49 samples of training data; while 12 levels of DOE are used for

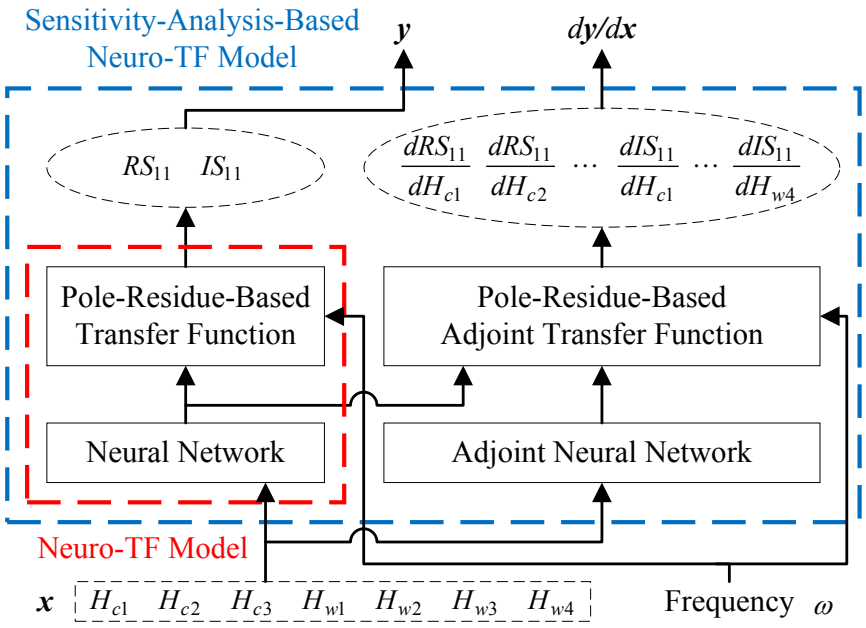


Figure 5.14: The structure of the proposed sensitivity-analysis-based neuro-TF model for the microwave cavity filter example. Seven geometrical parameters [H_{c1} H_{c2} H_{c3} H_{w1} H_{w2} H_{w3} H_{w4}] are used as the input variables of the microwave cavity filter model. The outputs of the original neuro-TF model $\boldsymbol{y}$ are the S-parameters, while the outputs of the adjoint neuro-TF model $d\boldsymbol{y}/d\boldsymbol{x}$ are the derivatives of S-parameters with respect to the geometrical parameters.

the testing data, i.e., a total of 144 samples of testing data. Each training and testing sample has 101 frequency data. The frequency range for this microwave cavity filter example is 630 MHz-730 MHz. After the proposed sensitivity-analysis-based neuro-TF model is developed using *NeuroModelerPlus* software, the final model used for design optimization only consists the original neuro-TF model.

In both cases, the orders N_k of transfer functions vary from 12 to 16 among different samples of geometrical parameters. Table 5.8 illustrates the comparison of training and testing errors in two cases using proposed pole-residue tracking

Table 5.7: Definition of Training and Testing Data for the Microwave Cavity Filter Example

Geometrical Parameters			Training Data (49 samples)			Testing Data (144 samples)		
			Min	Max	Step	Min	Max	Step
Case 1	Sensitivity Variables	H_{c1} (mm)	25.64	26.36	0.12	25.67	26.33	0.06
		H_{c2} (mm)	16.64	17.36	0.12	16.67	17.33	0.06
		H_{c3} (mm)	23.64	24.36	0.12	23.67	24.33	0.06
		H_{w1} (mm)	23.4	30.6	1.2	23.7	30.3	0.6
		H_{w2} (mm)	29.4	36.6	1.2	29.7	36.3	0.6
		H_{w3} (mm)	23.4	30.6	1.2	23.7	30.3	0.6
		H_{w4} (mm)	56.4	63.6	1.2	56.7	63.3	0.6
Case 2	Sensitivity Variables	H_{c1} (mm)	25.4	26.6	0.2	25.45	26.55	0.1
		H_{c2} (mm)	16.4	17.6	0.2	16.45	17.55	0.1
		H_{c3} (mm)	23.4	24.6	0.2	23.45	24.55	0.1
		H_{w1} (mm)	21	33	2	21.5	32.5	1
		H_{w2} (mm)	27	39	2	27.5	38.5	1
		H_{w3} (mm)	21	33	2	21.5	32.5	1
		H_{w4} (mm)	54	66	2	54.5	65.5	1

technique with sensitivity information and existing pole-residue tracking technique without sensitivity information [30]. In Case 1, since the geometrical step sizes between the training samples are relatively small, both methods obtain the correct pole splitting, resulting in good training and testing errors for both methods. In Case 2, when the geometrical step sizes between the training samples are large, part of the pole splitting is incorrect using the existing pole-residue tracking technique without sensitivity information [30]. This results in a relatively large error in both the preliminary training of the neural networks and the refinement training of the overall model. By using the proposed pole-residue tracking technique with the

Table 5.8: Comparisons of Different Pole-Residue Tracking Techniques for the Filter Example

	Pole-Residue Tracking Technique	Number of Hidden Neurons	ANN Training Error	ANN Testing Error	Overall Model Training Error	Overall Model Testing Error
Case 1	Existing Technique without Sensitivity	15	3.916%	4.324%	0.912%	0.973%
	Proposed Technique with Sensitivity	15	3.916%	4.324%	0.912%	0.973%
Case 2	Existing Technique without Sensitivity	15	5.825%	6.568%	2.186%	2.563%
	Existing Technique without Sensitivity	18	4.234%	9.562%	1.456%	3.365%
	Proposed Technique with Sensitivity	15	4.117%	4.986%	1.425%	1.683%

sensitivity information, we can obtain a better modeling accuracy even when the geometrical step sizes are large.

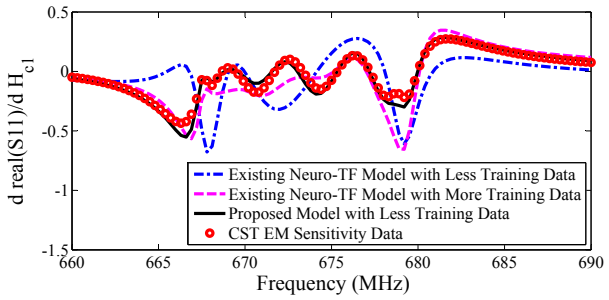
For comparison purpose, we also perform the modeling using existing SAANN technique [14]. Table 5.9 illustrates two cases of comparisons of training and testing errors among the SAANN Model using fewer data (49 training samples), existing neuro-TF model [30] using fewer data (49 training samples), existing neuro-TF model using more data (169 training samples), and the proposed model using fewer data (49 training samples). In Case 1, all methods obtain relatively small training and testing errors, because the geometrical parameters vary within a small range. In Case 2, when the range of geometrical parameters gets wider, the SAANN model and existing neuro-TF model trained with fewer training data cannot obtain good

Table 5.9: Comparisons of Different Modeling Methods for the Microwave Cavity Filter Example

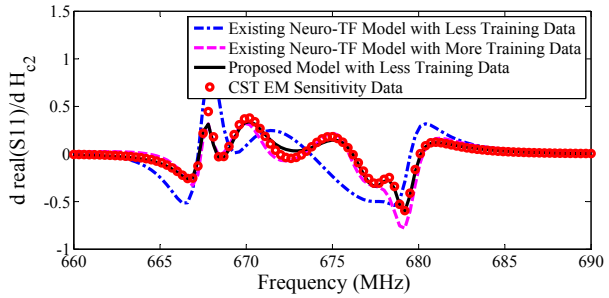
	Model Methods	Number of Hidden Neurons	Average Training Error	Average Testing Error
Case 1	SAANN Model Using 49 Samples of Training Data	50	0.952%	1.025%
	Existing Neuro-TF Model Using 49 Samples of Training Data	12	0.862%	1.124%
	Proposed Model Using 49 Samples of Training Data	15	0.912%	0.973%
Case 2	SAANN Model Using 49 Samples of Training Data	60	1.475%	15.567%
	Existing Neuro-TF Model Using 49 Samples of Training Data	15	1.532%	12.246%
	Existing Neuro-TF Model Using 169 Samples of Training Data	20	1.566%	1.912%
	Proposed Model Using 49 Samples of Training Data	15	1.425%	1.683%

testing accuracy even though they both maintain similarly good training accuracy. Fig. 5.15 shows the comparisons of derivatives of real part of S_{11} with respect to sensitivity variables H_{c1}, H_{c2} and H_{w1} by existing neuro-TF model using fewer data and more data, proposed model using fewer data and CST sensitivity analysis for one training sample $\boldsymbol{x} = [25.8\ 16.8\ 24.2\ 33\ 33\ 21\ 56]^T$ (mm). As shown in Fig. 5.15, the proposed model has more accurate derivative information after training process, which helps to obtain more accurate testing results.

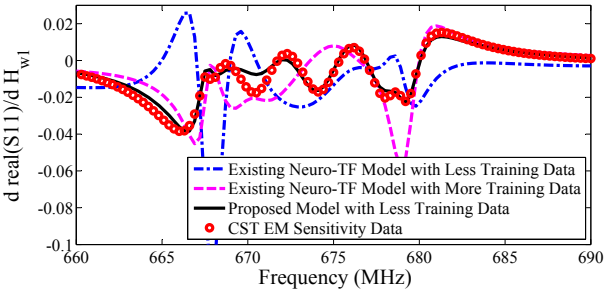
The comparisons of the outputs of the proposed model, existing neuro-TF model using different size of training data and EM data for three different testing geomet-



(a)



(b)



(c)

Figure 5.15: Comparison of the derivatives of the existing neuro-TF model using fewer and more data, the proposed model using fewer data and CST EM sensitivity data for this microwave cavity filter example: (a) derivatives of real parts of S_{11} with respect to sensitivity variable H_{c1}, (b) derivatives of real parts of S_{11} with respect to sensitivity variable H_{c2}, and (c) derivatives of real parts of S_{11} with respect to sensitivity variable H_{w1}. The derivatives of the proposed sensitivity-analysis-based neuro-TF model matches with EM sensitivity data from CST simulator much more closely than the existing method as shown in the figure.

rical parameters #1, #2 and #3 are shown in Fig. 5.16. The values of input geometrical variables for three different testing samples of microwave cavity filters are shown as follows.

Test geometrical sample

#1 : $\boldsymbol{x} = [25.45\ 17.15\ 24.15\ 28.5\ 34.5\ 28.5\ 61.5]^T$ (mm)

Test geometrical sample

#2 : $\boldsymbol{x} = [25.65\ 16.45\ 23.45\ 23.5\ 31.5\ 27.5\ 62.5]^T$ (mm)

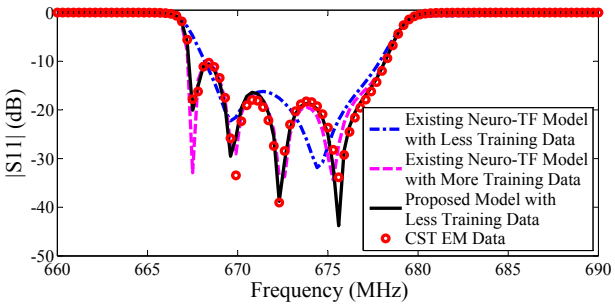
Test geometrical sample

#3 : $\boldsymbol{x} = [26.45\ 17.25\ 24.25\ 27.5\ 31.5\ 23.5\ 54.5]^T$ (mm)

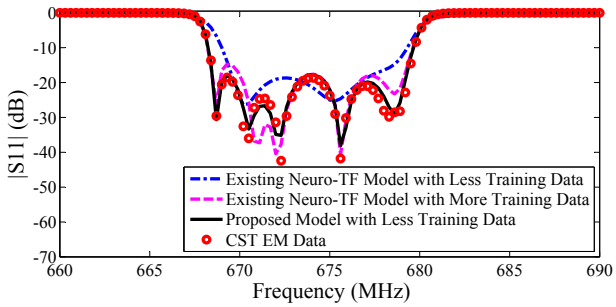
The magnitude of S_{21} of the proposed model and its comparison with EM data for the testing geometrical parameters #2 are shown in Fig. 5.17. Because the proposed model is trained very accurately (error around 1%) using real and imaginary parts of S-parameters, the dB values of S-parameters for the proposed model and the data can match well when the values of S-parameters are above -20 dB.

The three testing samples are not used during training process. As shown in Table 5.9 and Fig. 5.16, to achieve good model accuracy, fewer training data is needed for the proposed model than that needed for the existing neuro-TF model.

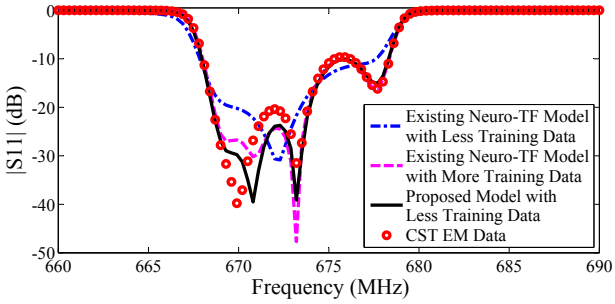
The development cost of the proposed neuro-TF model for this microwave cavity filter example is around 8.82 h, which consists of data generation cost (49 training samples) and training cost. For the development of existing neuro-TF model (169 geometrical samples of training data), the cost is around 23.4 h. Therefore, by reducing the size of training data using derivative information, our proposed technique can speedup the model training process over the existing technique.



(a)



(b)



(c)

Figure 5.16: Comparisons of S_{11} of the existing neuro-TF model using fewer and more data, the proposed model using fewer data and CST EM data: (a) test geometrical sample #1, (b) test geometrical sample #2, and (c) test geometrical sample #3 for this microwave cavity filter example. With the same size of data, the proposed model can obtain a better accuracy than the existing neuro-TF model as shown in the figure. To achieve good model accuracy, fewer training data is needed for the proposed model than that needed for the existing neuro-TF model.

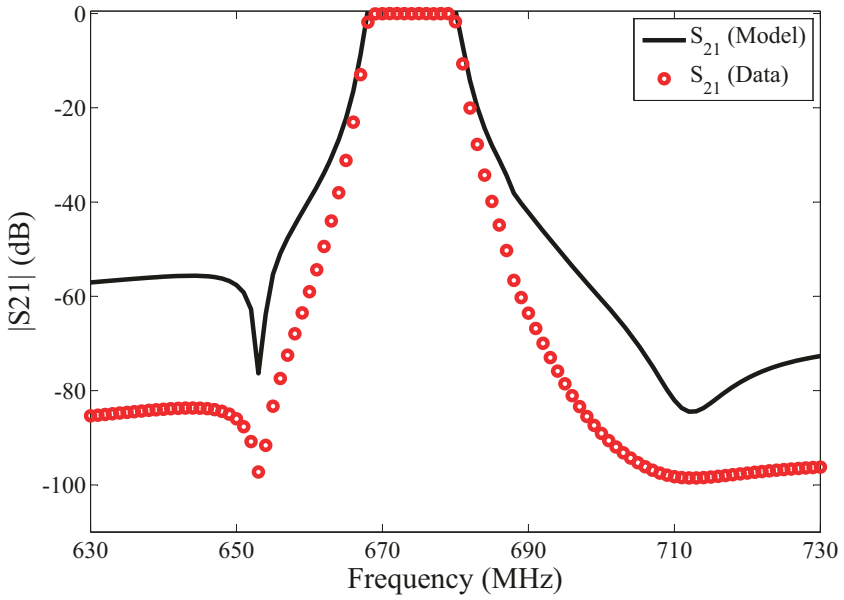


Figure 5.17: Comparisons of S_{21} (magnitude in decibels) of the trained proposed model and EM data of test geometrical sample #2 for the microwave cavity filter example.

5.4 Conclusion

In this chapter, a new sensitivity-analysis-based training technique for parametric neuro-transfer function modeling of microwave components has been proposed. The proposed model allows fast and accurate evaluation of EM behavior with respect to changes in geometrical parameters. In the proposed technique, both input-output response of the modeling problem and the sensitivity analysis information generated from EM simulators have been used in the model development. New formulations have been derived for calculating the second order derivatives for training the adjoint neuro-TF models. We have also proposed new formulations to extract the data

of pole/residues and their derivatives for the preliminary training of original and adjoint neural networks, respectively. The proposed technique can obtain accurate parametric models with less training data, compared to the previous neuro-TF modeling method. Consequently, fewer training data result in shorter time for model development using the proposed method. Further, with the same amount of training data, the proposed method has a better modeling accuracy compared to the previous method without sensitivity analysis. Once trained, the proposed models provide accurate and fast prediction of EM responses for high-level design with geometrical parameters as design variables.

Chapter 6

Conclusions and Future Research

6.1 Conclusions

In this thesis, a parametric modeling of EM behavior of microwave components using combined neural networks and pole-residue-based transfer functions has been developed. An advanced pole-residue tracking technique exploiting sensitivity analysis has been introduced. A further advancement of parametric modeling of microwave components using adjoint neural networks and pole-residue transfer functions with EM sensitivity analysis has also been proposed.

This thesis first proposes a new approach to parametric modeling of EM behavior of microwave components using combined neural networks and pole-residue-based transfer function. In the proposed method, a novel pole-residue tracking technique has been developed for pole-residue-based transfer function to deal with the problem of changing orders when geometrical parameters undergo large variations. With high dimensions of geometrical parameter space and large geometrical variations, the model obtained by the proposed method is more accurate and robust than

that obtained by the existing modeling method using rational transfer function formulation. The proposed method can also obtain good modeling accuracy even when the orders of the models are high. Once trained, the pole-residue-based neuro-TF model developed by the proposed method provides fast and accurate prediction of the EM responses w.r.t the geometrical variables. The developed pole-residue-based neuro-TF models can be also used for high-level optimization with geometrical parameters as design variables.

An advanced pole-residue tracking technique has been proposed to use the sensitivity analysis information to obtain more reliable pole splitting than the existing pole-residue tracking technique. The proposed technique has taken advantages of sensitivity information to split one pole into two separate new poles to achieve the increase of the order of the transfer function and ultimately form transfer functions of a constant order over the entire region of geometrical parameters. The proposed technique has addressed the challenges of pole-residue tracking when training data are limited.

A further advancement has been proposed by utilizing sensitivity analysis into parametric modeling of microwave components with adjoint neural networks and pole-residue transfer functions. In the proposed technique, both input-output response of the modeling problem and the sensitivity analysis information generated from EM simulators have been used in the model development. New formulations have been derived for calculating the second order derivatives for training the adjoint neuro-TF models. We have also proposed new formulations to extract the data of pole/residues and their derivatives for the preliminary training of original and

adjoint neural networks, respectively. Consequently, fewer training data result in shorter time for model development using the proposed method. Further, with the same amount of training data, the proposed method has a better modeling accuracy compared to the previous method without sensitivity analysis.

6.2 Future Research

Some of the future directions based on the expansion of the proposed parametric modeling techniques have been identified in this thesis as follows

- One of the future directions is to apply the proposed parametric modeling technique to the modeling of multi-physics problems. For high performance microwave design, the consideration of operations in a real world multi-physics environment is often required. Multi-physics simulations offer comprehensive solutions capable of performing bi-directional coupled analysis between different physics domain, such as EM, thermal and structural mechanics. The multi-physics simulation is very computationally expensive because it involves multiple domains and usually deals with the deformed non-ideal structures. This problem becomes even more challenging when repetitive simulations of a given structure are required due to adjustments of the values of design parameters. By utilizing our proposed technique, we can establish accurate parametric models to represent the output responses from multi-physics simulations. The developed parametric model can be thus used in to the high level circuit and system analysis and design optimizations.

- Another future direction is to expand the proposed parametric modeling for producing feature parameters to guide surrogate-based EM optimizations for the microwave filter design. In each iteration of the surrogate-based EM optimization, a parametric model using pole-residue-based neuro-transfer functions is established to represent the EM behaviors. We can extract the poles out of the parametric model to be considered as a set of feature frequencies to guide the surrogate optimization in each iteration. We can modify the conventional design objective functions into feature-based objective functions so that the filter designer can focus on moving the feature frequencies into pass-band and forcing the surrogate outputs satisfying the design specifications simultaneously. Guided with the feature frequencies, EM optimizations have a better chance of avoiding the local minima and reach optimal EM solutions faster.

- As a further research, the proposed parametric modeling technique can be also applied to yield estimations and optimizations. Using the proposed technique, we can develop the surrogate model representing the EM behavior of the microwave components in a relatively large range. After an accurate parametric model is developed, we can perform Monte-Carlo analysis to do the yield estimation using the developed model within the geometrical region. The developed model can also provide sensitivity information which can be further formulated into the optimization directions for gradient-based yield optimizations.

- Another further work is to expand the proposed parametric modeling tech-

nique addressing parametric modeling problems with large geometrical range. We can produce a large number of parametric models using the proposed technique as sub-models to finally form a large range overall model. A specific set of connection functions can be formulated to smoothly link all the sub-models. The connection functions need to be developed to obtain the continuity of not only the outputs but also the first order derivatives with respect to the inputs of the overall model. Since the overall model has a very large geometrical range, the design optimization with respect to geometrical parameters as design variables can be performed in a large neighborhood using the developed overall model.

References

[1] M. B. Steer, J. W. Bandler, and C. M. Snowden, "Computer-aided design of RF and microwave circuits and systems," *IEEE Trans. Microw. Theory Techn.*, vol. 50, no. 3, pp. 996-1005, Mar. 2002.

[2] J. E. Rayas-Sanchez, "EM-based optimization of microwave circuits using artificial neural networks: The state-of-the-art," *IEEE Trans. Microw. Theory Techn.*, vol. 52, no. 1, pp. 420-435, Jan. 2004.

[3] Q. J. Zhang, K. C. Gupta, and V. K. Devabhaktuni, "Artificial neural networks for RF and microwave design: from theory to practice," *IEEE Trans. Microw. Theory Techn.,* vol. 51, no. 4, pp. 1339-1350, Apr. 2003.

[4] A. H. Zaabab, Q. J. Zhang, and M. S. Nakhla, "A neural network modeling approach to circuit optimization and statistical design," *IEEE Trans. Microw. Theory Techn.,* vol. 43, no. 6, pp. 1349-1358, June 1995.

[5] V. Rizzoli, A. Costanzo, D. Masotti, A. Lipparini, and F. Mastri, "Computer-aided optimization of nonlinear microwave circuits with the aid of electromagnetic simulation," *IEEE Trans. Microw. Theory Techn.*, vol. 52, no. 1, pp. 362-377, Jan. 2004.

[6] P. Burrascano, S. Fiori, and M. Mongiardo, "A review of artificial neural networks applications in microwave computer-aided design," *Int. J. RF Microw. Comput.-Aided Eng.*, vol. 9, no. 3, pp. 158-174, May 1999.

[7] S. K. Mandal, S. Sural, and A. Patra, "ANN- and PSO-based synthesis of on-chip spiral inductors for RF ICs," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.,* vol. 27, no. 1, pp. 188-192, Jan. 2008.

[8] T. Liu, S. Boumaiza, and F. M. Ghannouchi, "Dynamic behavioral modeling of 3G power amplifiers using real-valued time-delay neural networks," *IEEE Trans. Microw. Theory Techn.,* vol. 52, no. 3, pp. 1025-1033, Mar. 2004.

[9] M. Isaksson, D. Wisell, and D. Ronnow, "Wide-band dynamic modeling of power amplifiers using radial-basis function neural networks," *IEEE Trans. Microw. Theory Techn.,* vol. 53, no. 11, pp. 3422-3428, Nov. 2005.

[10] J. Wood, D. E. Root, and N. B. Tufillaro, "A behavioral modeling approach to nonlinear model-order reduction for RF/microwave ICs and systems," *IEEE Trans. Microw. Theory Techn.,* vol. 52, no. 9, pp. 2274-2284, Sep. 2004.

[11] Y. Cao, S. Reitzinger, and Q. J. Zhang, "Simple and efficient high-dimensional parametric modeling for microwave cavity filters using modular neural network," *IEEE Microw. Wireless Compon. Lett.,* vol. 21, no. 5, pp. 258-260, May 2011.

[12] Q. J. Zhang and K. C. Gupta, *Neural Networks for RF and Microwave Design.* Norwood, MA: Artech House, 2000.

[13] J. Xu, M. C. E. Yagoub, R. Ding, and Q. J. Zhang, "Exact adjoint sensitivity analysis for neural-based microwave modeling and design," *IEEE Trans. Microw. Theory Techn.*, vol. 51, no. 1, pp. 226-237, Jan. 2003.

[14] S. A. Sadrossadat, Y. Cao, and Q. J. Zhang, "Parametric Modeling of Microwave Passive Components Using Sensitivity-Analysis-Based Adjoint Neural-Network Technique," *IEEE Trans. Microw. Theory Techn.*, vol. 61, no. 5, pp. 1733-1747, May 2013.

[15] N. K. Nikolova, J. W. Bandler, and M. H. Bakr, "Adjoint techniques for sensitivity analysis in high-frequency structure CAD," *IEEE Trans. Microw. Theory Techn.*, vol. 52, no. 1, pp. 403-419, Jan. 2004.

[16] V. K. Devabhaktuni, B. Chattaraj, M. C. E. Yagoub, and Q. J. Zhang, "Advanced microwave modeling framework exploiting automatic model generation, knowledge neural networks, and space mapping," *IEEE Trans. Microw. Theory Techn.*, vol. 51, no. 7, pp. 1822-1833, Jul. 2003.

[17] J. W. Bandler, M. A. Ismail, J. E. Rayas-Sanchez, and Q. J. Zhang, "Neuromodeling of microwave circuits exploiting space-mapping technology," *IEEE Trans. Microw. Theory Techn.*, vol. 47, no. 12, pp. 2417-2427, Dec. 1999.

[18] J. E. Rayas-Sanchez and V. Gutierrez-Ayala, "EM-based Monte Carlo analysis and yield prediction of microwave circuits using linear-input neural-output space mapping," *IEEE Trans. Microw. Theory Techn.*, vol. 54, no. 12, pp. 4528-4537, Dec. 2006.

[19] J. W. Bandler, Q. S. Cheng, S. A. Dakroury, A. S. Mohamed, M. H. Bakr, K. Madsen, and J. Sondergaard, "Space mapping: The state of the art," *IEEE Trans. Microw. Theory and Techn.,* vol. 52, no. 1, pp. 337-361, Jan. 2004.

[20] S. Koziel, Q. S. Cheng, and J. W. Bandler, "Space mapping," *IEEE Microw. Mag.,* vol. 9, no. 6, pp. 105-122, Dec. 2008.

[21] F. Feng, V. M. R. Gongal-Reddy, S. Zhang, and Q. J. Zhang, "Recent advances in space mapping approach to EM optimization," in *Proc. AsiaPacific Microw. Conf.*, Nanjing, China, Dec. 2015, pp. 1-3.

[22] A. Garcia-Lamperez, S. Llorente-Romano, M. Salazar-Palma, and T. K. Sarkar, "Efficient electromagnetic optimization of microwave filters and multiplexers using rational models," *IEEE Trans. Microw. Theory Techn.*, vol. 52, no. 2, pp. 508-521, Feb. 2004.

[23] A. Garcia-Lamperez and M. Salazar-Palma, "Multilevel aggressive space mapping applied to coupled-resonator filters," in *IEEE MTT-S Int. Microw. Symp. Dig.,* San Francisco, CA, May. 2016, pp. 1-4.

[24] M. Sans, J. Selga, P. Velez, A. Rodriguez, J. Bonache, V.E. Boria, and F. Martin, "Automated design of common-mode suppressed balanced wideband bandpass filters by means of aggressive space mapping," *IEEE Trans. Microw. Theory Techn.*, vol. 63, no. 12, pp. 3896-3908, Dec. 2015.

[25] F. Feng, C. Zhang, V. M. R. Gongal-Reddy, Q. J. Zhang, and J. Ma, "Parallel space-mapping approach to EM optimization," *IEEE Trans. Microw. Theory Techn.*, vol. 62, no. 5, pp. 1135-1148, Apr. 2014.

[26] X. Ding, V. K. Devabhaktuni, B. Chattaraj, M. C. E. Yagoub, M. Deo, J. Xu, and Q. J. Zhang, "Neural-network approaches to electromagnetic based modeling of passive components and their applications to high frequency and high-speed nonlinear circuit optimization," *IEEE Trans. Microw. Theory Techn.*, vol. 52, no. 1, pp. 436-449, Jan. 2004.

[27] V. M. R. Gongal-Reddy, F. Feng, and Q. J. Zhang, "Parametric modeling of millimeter-wave passive components using combined neural networks and transfer functions," in *Global Symposium On Millimeter Waves (GSMM)*, Montreal, QC, Canada, May 2015, pp. 1-3.

[28] Y. Cao, G. Wang, and Q. J. Zhang, "A new training approach for parametric modeling of microwave passive components using combined neural networks and transfer functions," *IEEE Trans. Microw. Theory Techn.*, vol. 57, no. 11, pp. 2727-2742, Nov. 2009.

[29] Z. Guo, J. Gao, Y. Cao, and Q. J. Zhang, "Passivity enforcement for passive component modeling subject to variations of geometrical parameters using neural networks," in *IEEE MTT-S Int. Microw. Symp. Dig.,* Montreal, QC, Canada, Jun. 2012, pp. 1-3.

[30] F. Feng, C. Zhang, J. Ma, and Q. J. Zhang, "Parametric modeling of EM behavior of microwave components using combined neural networks and pole-residue-based transfer functions," *IEEE Trans. Microw. Theory Techn.*, vol. 64, no. 1, pp. 60-77, Jan. 2016.

[31] F. Feng, V. M. R. Gongal-Reddy, C. Zhang, J. Ma, and Q. J. Zhang, "Parametric Modeling of Microwave Components Using Adjoint Neural Networks and Pole-Residue Transfer Functions With EM Sensitivity Analysis," *IEEE Trans. Microw. Theory Techn.*, in press, Feb. 2017.

[32] F. Feng and Q. J. Zhang, "Parametric modeling using sensitivity-based adjoint neuro-transfer functions for microwave passive components," in *IEEE MTT-S Int. Conf. Numerical Electromagnetic and Multiphysics Modeling and Optim.,* Ottawa, Canada, pp. 1-3, Aug. 2015.

[33] D. Schreurs, J. Verspecht, S. Vandenberghe, and E. Vandamme, "Straightforward and accurate nonlinear device model parameter-estimation method based on vectorial large-signal measurements," *IEEE Trans. Microw. Theory Techn,* vol. 50, no. 10, pp. 2315-2319, Oct. 2002.

[34] W. Zhang, S. Yan, F. Feng, and Q. J. Zhang, "Fast and simple technique for computing circuit noise figure from component noise model using artificial neural network," in *IEEE MTT-S Int. Conf. Numerical Electromagnetic and Multiphysics Modeling and Optim.*, Ottawa, Canada, Aug. 2015, pp. 1-3.

[35] F. Wang and Q. J. Zhang, "Knowledge based neural models for microwave design," *IEEE Trans. Microw. Theory Techn.,* vol. 45, no. 12, pp. 2333-2343, Dec. 1997.

[36] G. L. Creech, B. J. Paul, C. D. Lesniak, T. J. Jenkins, and M. C. Calcatera "Artificial neural networks for fast and accurate EM-CAD of microwave circuits," *IEEE Trans. Microwave Theory Techn.,* vol. 45, no. 5, pp. 794-802, May 1997.

[37] 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. Microw. Theory Techn.,* vol. 45, no. 5, pp. 712-723, May 1997.

[38] Q. J. Zhang and M. S. Nakhla, "Signal integrity analysis and optimization of VLSI interconnects using neural network models," *IEEE Int. Circuits Syst. Symp.,* London, England, May 1994, pp. 459-462.

[39] J. Bandler, M. Ismail, J. Rayas-Sanchez, and Q. J. Zhang, "New directions in model development for RF/microwave components utilizing artificial neural networks and space mapping," *IEEE AP-S Int. Symp. Digest,* Orlando, FL, pp. 2572-2575, Jul. 1999.

[40] P. M. Watson and K. C. Gupta, "EM-ANN models for microstrip vias and interconnects in multilayer circuits," *IEEE Trans. Microw. Theory Techn.,* vol. 44, no. 12, pp. 2495-2503, Dec. 1996.

[41] Y. Cao and G. Wang, "A wideband and scalable model of spiral inductors using space-mapping neural network," *IEEE Trans. Microw. Theory Techn.,* vol. 55, no. 12, pp. 2473-2480, Dec. 2007.

[42] V. B. Litovski, J. I. Radjenovic, Z. M. Mrcarica, and S. L. Milenkovic, "MOS transistor modeling using neural network," *Elect. Lett.,* vol. 28, no. 18, pp. 1766-1768, Aug. 1992.

[43] V. K. Devabhaktuni, C. Xi, and Q. J. Zhang, "A neural network approach to the modeling of heterojunction bipolar transistors from S-parameter data," *Proc.* 28^{th} *European Microw. Conf.,* Amsterdam, Netherlands, Oct. 1998, pp. 306-311.

[44] M. Vai and S. Prasad, "Qualitative modeling heterojunction bipolar transistors for optimization: A neural network approach," *Proc. IEEE/Cornell Conf. Adv. Concepts in High Speed Semiconductor Dev. and Circuits,* pp. 219-227, Aug. 1993.

[45] K. Shirakawa, M. Shimizu, N. Okubo, and Y. Daido, "Structural determination of multilayered large signal neural-network HEMT model," *IEEE Trans. Microw. Theory Techn.,* vol. 46, no. 10, pp. 1367-1375, Oct. 1998.

[46] G. Fedi, S. Manetti, G. Pelosi, and S. Selleri, "Design of cylindrical posts in rectangular waveguide by neural network approach," *IEEE AP-S Int. Symp. Digest,* Salt Lake City, UT, Jul. 2000, pp. 1054-1057.

[47] Q. J. Zhang, G. Wilson, R. Venkatachalam, A. Sarangan, J. Williamson, and F. Wang, "Ultra fast neural models for analysis of electro/optical interconnects," *Proc. IEEE Electronic Components and Techn. Conf.* San Jose, CA, May 1994, pp. 1134-1137.

[48] M. H. Bakr, J. W. Bandler, M. A. Ismail, J. E. Rayas-Sanchez, and Q. J. Zhang, "Neural space mapping EM optimization of microwave structures," *IEEE MTT-S Int. Microw. Symp. Dig.,* Boston, MA, Jun. 2000, pp. 879-882.

[49] H. Sharma and Q. J. Zhang, "Transient electromagnetic modeling using recurrent neural networks," *IEEE MTT-S Int. Microw. Symp. Dig.,* San Francisco, CA, Jun. 2005, pp. 1597-1600.

[50] H. Kabir, Y. Wang, M. Yu, and Q. J. Zhang, "Neural network inverse modeling and applications to microwave filter design," *IEEE Trans. Microw. Theory Techn.,* vol. 56, no. 4, pp. 867-879, Apr. 2008.

[51] Y. Fang, M. C. E. Yagoub, F. Wang, and Q. J. Zhang, "A new macromodeling approach for nonlinear microwave circuits based on recurrent neural networks," *IEEE Trans. Microw. Theory Techn.,* vol. 48, no. 12, pp. 2335-2344, Dec. 2000.

[52] C. Christodoulou, A. E. Zooghby, and M. Georgiopoulos, "Neural network processing for adaptive array antennas," *IEEE-APS Int. Symp.,* Orlando, FL, July 1999, pp.2584-2587.

[53] P. M. Watson and K. C. Gupta, "Design and optimization of CPW circuits using EM-ANN models for CPW components," *IEEE Trans. Microw. Theory Techn.,* vol. 45, no. 12, pp. 2515-2523, Dec. 1997.

[54] X. Ding, J. Xu, M. C. E. Yagoub, and Q. J. Zhang, "A combined state space formulation/equivalent circuit and neural network technique for modeling of embedded passives in multilayer printed circuits," *J. of the Applied Computational Electromagnetics Society,* vol. 18, 2003.

[55] Y. Cao, L. Simonovich, and Q. J. Zhang, "A broadband and parametric model of differential via holes using space-mapping neural network," *IEEE Microw. Wireless Compon. Lett.,* vol. 19, no. 9, pp. 533-535, Sep. 2009.

[56] J. Xu, M. C. E. Yagoub, R. Ding, and Q. J. Zhang, "Neural based dynamic modeling of nonlinear microwave circuits," *IEEE Trans. Microw. Theory Techn.,* vol. 50, no. 12, pp. 2769-2780, Dec. 2002.

[57] Y. Cao, J. J. Xu, V. K. Devabhaktuni, R. T. Ding, and Q. J. Zhang, "An adjoint dynamic neural network technique for exact sensitivities in nonlinear transient modeling and high-speed interconnect design," in *IEEE MTT-S Int. Microw. Symp. Dig.,* PA, Philadelphia, Jun. 2003, pp. 165-168.

[58] B. O'Brien, J. Dooley, and T. J. Brazil, "RF power amplifier behavioral modeling using a globally recurrent neural network," in *IEEE MTT-S Int. Microw. Symp. Dig.,* San Francisco, CA, Jun. 2006, pp. 1089-1092.

[59] D. Schreurs, J. Wood, N. Tufillaro, L. Barford, and D. E. Root, "Construction of behavioral models for microwave devices from time domain large signal measurements to speed up high-level design simulations," *Int. J. RF Microw. Comput.-Aided Eng.,* vol. 13, no. 1, pp. 54-61, Jan. 2003.

[60] H. Sharma and Q. J. Zhang, "Automated time domain modeling of linear and nonlinear microwave circuits using recurrent neural networks," *Int. J. RF Microw. Comput.-Aided Eng.,* vol. 18, no. 3, pp. 195-208, May 2008.

[61] I. A. Maio, I. S. Stievano, and F. G. Canavero, "NARX approach to black-box modeling of circuit elements," in *Proc. IEEE Int. Symp. Circuits Syst.,* Monterey, CA, Jun. 1998, pp. 411-414.

[62] V. Rizzoli, A. Neri, D. Masotti, and A. Lipparini, "A new family of neural network-based bidirectional and dispersive behavioral models for nonlinear RF/microwave subsystems," *Int. J. RF Microw. Comput.- Aided Eng.,* vol. 12, no. 1, pp. 51-70, Jan. 2002.

[63] I. S. Stievano, I. A. Maio, and F. G. Canavero, "Parametric macromodels of digital I/O ports," *IEEE Trans. Adv. Packag.,* vol. 25, no. 5, pp. 255-264, May 2002.

[64] Y. Cao, R. T. Ding, and Q. J. Zhang, "State-space dynamic neural network technique for high-speed IC applications: Modeling and stability analysis," *IEEE Trans. Microw. Theory Techn.,* vol. 54, no. 6, pp. 2398-2409, Jun. 2006.

[65] Y. Cao, R. T. Ding, and Q. J. Zhang, "A new nonlinear transient modeling technique for high-speed integrated circuit applications based on state-space dynamic neural network," *IEEE MTT-S Int. Microw. Symp. Dig.,* Fort Worth, TX, Jun. 2004, pp. 1553-1556.

[66] J. M. Zamarreno, P. Vega, L. D. Garcia, and M. Francisco, "State-space neural network for modeling, prediction and control, *Contr. Eng. Practice,* vol. 8, no. 9, pp. 1063-1075, Sep. 2000.

[67] P. Gil, A. Dourado, and J. O. Henriques, "State space neural networks and the unscented Kalman filter in online nonlinear system identification," *IASTED Int. Conf. Intell. Syst. Contr.,* Tampa, FL, Nov. 2001, pp. 337-342.

[68] V. K. Devabhakruni, M. C. E. Yagoub, and Q. J. Zhang, "A robust algorithm for automatic development of neural-network models for microwave applications," *IEEE Trans. Microw. Theory Techn.,* vol. 49, no. 12, pp. 2282-2291, Dec. 2001.

[69] W. Na and Q. J. Zhang, "Automated knowledge-based neural network modeling for microwave applications," *IEEE Microw. Wireless Compon. Lett.,* vol. 24, no. 7, pp. 499-501, Jul. 2014.

[70] W. Na, F. Feng, C. Zhang, and Q. J, Zhang, "A unified automated parametric modeling algorithm using knowledge-based neural network and L_1 optimization," *IEEE Trans. Microw. Theory Techn.*, vol. 65, no. 3, pp. 729-745, Mar. 2017.

[71] H. Kabir, L. Zhang, M. Yu, P. H. Aaen, J. Wood, and Q. J. Zhang, "Smart modeling of microwave device," *IEEE Microw. Mag.*, vol. 11, no. 3, pp. 105-118, May 2010.

[72] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Math. Control Signals Systems,* vol. 2, pp. 303-314, 1989.

[73] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks,* vol. 2, pp. 359-366, 1989.

[74] T. Y. Kwok, and D. Y. Yeung, "Constructive algorithms for structure learning in feedforward neural networks for regression problems," *IEEE Trans. Neural Networks,* vol. 8, no. 3, pp. 630-645, May 1997.

[75] R. Reed, "Pruning algorithms - a Survey," *IEEE Trans. Neural Networks,* vol. 4, no. 5, pp. 740-747, Sep. 1993.

[76] A. Krzyzak, and T. Linder, "Radial basis function networks and complexity regularization in function learning," *IEEE Trans. Neural Networks,* vol. 9, no. 2, pp. 247-256, Mar. 1998.

[77] J. de Villiers, and E. Barnard, "Backpropagation neural nets with one and two hidden layers," *IEEE Trans. Neural Networks,* vol. 4, no. 1, pp. 136-141, Jan. 1993.

[78] S. Tamura, and M. Tateishi, "Capabilities of a four-layered feedforward neural network: four layer versus three," *IEEE Trans. Neural Networks,* vol. 8, no. 2, pp. 251-255, Mar. 1997.

[79] J. A. Garcia, et al., "Modeling MESFET's and HEMT's intermodulation distortion behavior using a generalized radial basis function network," *Int. Journal of RF and Microwave CAE,* Special Issue on Applications of ANN to RF and Microwave Design, Vol. 9, pp. 261-276, 1999.

[80] Q. H. Zhang, "Using wavelet network in nonparametric estimation," *IEEE Trans. Neural Networks,* vol. 8, no. 2, pp. 227-236, Mar. 1997.

[81] Q. H. Zhang and A. Benvensite, "Wavelet networks," *IEEE Trans. Neural Networks,* vol. 3, no. 6, pp. 889-898, Nov. 1992.

[82] Y. C. Pati and P. S. Krishnaprasad, "Analysis and synthesis of feedforward neural networks using discrete affine wavelet transformations," *IEEE Trans. Neural Networks,* vol. 4, no. 1, pp. 73-85, Jan. 1993.

[83] J. Aweya, Q. J. Zhang, and D. Montuno, "A Direct Adaptive Neural Controller for Flow Control in Computer Networks," *IEEE Int. Conf. Neural Networks,* Anchorage, Alaska, pp. 140-145, May 1998.

[84] J. Aweya, Q. J. Zhang, and D. Montuno, "Modelling and Control of Dynamic Queues in Computer Networks using Neural Networks," *IASTED Int. Conf. Intelligent Syst. Control*, Halifax, Canada, pp. 144-151, June 1998.

[85] L. H. Tsoukalas, and R. E. Uhrig, *Fuzzy and Neural Approaches in Engineering*, NY: Wiley-Interscience, 1997.

[86] J. W. Bandler, Q. S. Cheng, N. K. Nikolova, and M. A. Ismail, "Implicit space mapping optimization exploiting preassigned parameters," *IEEE Trans. Microw. Theory Tech.,* vol. 52, no. 1, pp. 378-385, Jan. 2004.

[87] J. W. Bandler, M. A. Ismail, and J. E. Rayas-Sanchez, "Expanded space-mapping EM-based design framework exploiting preassigned parameters," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.,* vol. 49, no. 12, pp. 1833-1838, Dec. 2002.

[88] S. Koziel, J. W. Bandler, and K. Madsen, "Space mapping with adaptive response correction for microwave design optimization," *IEEE Trans. Microw. Theory Techn.*, vol. 57, no. 2, pp. 478-486, Feb. 2009.

[89] J. W. Bandler, D. M. Hailu, K. Madsen, and F. Pedersen, "A space mapping interpolating surrogate algorithm for highly optimized EM-based design of microwave devices," *IEEE Trans. Microw. Theory Techn.,* vol. 52, no. 11, pp. 2593-2600, Nov. 2004.

[90] R. B. Ayed, J. Gong, S. Brisset, F. Gillon, and P. Brochet, "Three-level output space mapping strategy for electromagnetic design optimization," *IEEE Trans. Magn.*, vol. 48, no. 2, pp. 671-674, Feb. 2012.

[91] L. Zhang, J. Xu, M. C. E. Yagoub, R. Ding, and Q. J. Zhang, "Efficient analytical formulation and sensitivity analysis of neuro-space mapping for nonlinear microwave device modeling," *IEEE Trans. Microw. Theory Techn.*, vol. 53, no. 9, pp. 2752-2767, Sep. 2005.

[92] M. H. Bakr, J. W. Bandler, M. A. Ismail, J. E. Rayas-Sanchez, and Q. J. Zhang, "Neural space-mapping optimization for EM-based design," *IEEE Trans. Microw. Theory Techn.,* vol. 48, no. 12, pp. 2307-2315, Dec. 2000.

[93] V. Gutierrez-Ayala and J. E. Rayas-Sanchez, "Neural input space mapping optimization based on nonlinear two-layer perceptrons with optimized nonlinearity," *Int. J. RF Microw. Comput.-Aided Eng.,* vol. 20, no. 5, pp. 512-526, Sep. 2010.

[94] F. Feng and Q. J. Zhang, "Neural space mapping optimization for EM design," in *Proc. Asia-Pacific Microw. Conf.*, Nanjing, China, Dec. 2015, pp. 1-3.

[95] D. Gorissen, L. Zhang, Q. J. Zhang, and T. Dhaene, "Evolutionary neuro-space mapping technique for modeling of nonlinear microwave devices," *IEEE Trans. Microw. Theory Techn.*, vol. 59, no. 2, pp. 213-229, Feb. 2011.

[96] S. Koziel, J. W. Bandler, and K. Madsen, "A space mapping framework for engineering optimization: theory and implementation," *IEEE Trans. Microw. Theory Techn.*, vol. 54, no. 10, pp. 3721-3730, Oct. 2006.

[97] S. Koziel, J. W. Bandler, and Q. S. Cheng, "Tuning space mapping design framework exploiting reduced electromagnetic models," *IET Microw. Antennas Propag.*, vol. 5, no. 10, pp. 1219-1226, Jul. 2011.

[98] J. Meng, S. Koziel, J. W. Bandler, M. H. Bakr, and Q. S. Cheng, "Tuning space mapping: A novel technique for engineering design optimization," in *IEEE MTT-S Int. Microw. Symp. Dig.,*, Atlanta, Georgia, Jun. 2008, pp. 991-994.

[99] L. Zhang, P. H. Aaen, and J. Wood, "Portable space mapping for efficient statistical modeling of passive components," *IEEE Trans. Microw. Theory Techn.*, vol. 60, no. 3, pp. 441-450, Mar. 2012.

[100] C. Zhang, F. Feng, and Q. J. Zhang, "EM optimization using coarse and fine mesh space mapping," in *Proc. Asia-Pacific Microw. Conf.,* Seoul, Korea, Dec. 2013, pp. 824-826.

[101] F. Feng, C. Zhang, V. M. R. Gongal-Reddy, and Q. J. Zhang, "Knowledge-based coarse and fine mesh space mapping approach to EM optimization," in *Int. Conf. Numerical Electromagnetic Modeling and Optimization,* Pavia, Italy, May. 2014, pp. 1-4.

[102] S. Koziel, J. W. Bandler, and Q. S. Cheng, "Constrained parameter extraction for microwave design optimization using implicit space mapping," *IET Microw., Antennas & Prop.,* vol. 5, no. 10, pp. 1156-1163, Jul. 2011.

[103] S. Koziel, S. Ogurtsov, J. W. Bandler, and Q. S. Cheng, "Reliable space-mapping optimization integrated with EM-based adjoint sensitivities," *IEEE Trans. Microw. Theory Tech.,* vol. 61, no. 10, pp. 3493-3502, Oct. 2013.

[104] S. Koziel, Q. S. Cheng, and J. W. Bandler, "Fast EM modeling exploiting shape-preserving response prediction and space mapping," *IEEE Trans. Microw. Theory Tech.,* vol. 62, no. 3, pp. 399-407, Mar. 2014.

[105] J. W. Bandler, Q. S. Cheng, D. H. Gebre-Mariam, K. Madsen, F. Pedersen, and J. Sondergaard, "EM-based surrogate modeling and design exploiting im-

plicit, frequency and output space mappings," in *IEEE MTT-S Int. Microwave Symp. Dig.,* Philadelphia, PA, Jun. 2003, pp. 1003-1006

[106] N. M. Alexandrov, J. E. Dennis Jr., R. M. Lewis, and V. Torczon, "A trust-region framework for managing the use of approximation models in optimizations," *Struct. Optim.,* vol. 15, no. 1, pp. 16-23, Feb. 1998.

[107] N. M. Alexandrov and R. M. Lewis, "An overview of first-order model management for engineering optimization," *Optim. Eng.,* vol. 2, no. 4, pp. 413-430, Dec. 2001.

[108] N. M. Alexandrov, R. M. Lewis, C. R. Gumbert, L. L. Green, and P. A. Newman, "Approximation and model management in aerodynamic optimization with variable-fidelity models," *J. of Aircraft,* vol. 38, no. 6, pp. 1093-1101, Dec. 2001.

[109] R. Trobec, M. Vajtersic, and P. Zinterhof, *Parallel Computing: Numerics, Applications, and Trends,* Springer Dordrecht Heidelberg London New York, 2009.

[110] L. Zhang, Y. Cao, S. Wan, H. Kabir, and Q. J. Zhang, "Parallel automatic model generation technique for microwave modeling," in *IEEE MTT-S Int. Microw. Symp. Dig.,* Honolulu, HI, Jun. 2007, pp. 103-106.

[111] F. Feng, V. M. R. Gongal-Reddy, C. Zhang, W. Na, S. Zhang, and Q. J. Zhang, "Recent advances in parallel EM optimization approaches," in *IEEE*

MTT-S Int. Conf. Microw. Millimeter Wave Technology, Beijing, China, Jun. 2016, pp. 1-3.

[112] F. Feng, C. Zhang, S. Zhang, V. M. R. Gongal-Reddy, and Q. J. Zhang, "Parallel EM optimization approach to microwave filter design using feature assisted neuro-transfer functions," in *IEEE MTT-S Int. Microw. Symp. Dig.*, San Francisco, CA, May. 2016, pp. 1-3.

[113] J. Zhang, K. Ma, F. Feng, Z. Zhao, W. Zhang, and Q. J. Zhang, "Distributed parallel computing technique for EM modeling," in *IEEE MTT-S Int. Conf. Numerical Electromagnetic and Multiphysics Modeling and Optim.*, Ottawa, Canada, Aug. 2015, pp. 1-3.

[114] J. Zhang, K. Ma, F. Feng, and Q. J. Zhang, "Parallel gradient-based local search accelerating particle swarm optimization for training microwave neural network models," in *IEEE MTT-S Int. Microw. Symp. Dig.*, Phoenix, AZ, May. 2015, pp. 1-3.

[115] V. M. R. Gongal-Reddy, S. Zhang, Y. Cao, and Q. J. Zhang, "Efficient design optimization of microwave circuits using parallel computational methods," in *Proc. 7th Eur. Microw. Integr. Circuits Conf.,* Amsterdam, The Netherlands, Oct. 2012, pp. 254-257.

[116] S. Koziel and J. W. Bandler, "Space mapping with distributed fine model evaluation for optimization of microwave structures and devices," in *IEEE MTT-S Int. Microw. Symp. Dig.,* Atlanta, GA, Jun. 2008, pp. 1377- 1380.

[117] S. Koziel and J. W. Bandler, "Distributed fine model evaluation for rapid space-mapping optimisation of microwave structures," *IET Microw. Antennas Propag.*, vol. 3, no. 5, pp. 798-807, Aug. 2009.

[118] C. Zhang, F. Feng, V. M. R. Gongal-Reddy, Q. J. Zhang, and J. W. Bandler, "Cognition-driven formulation of space mapping for equal-ripple optimization of microwave filters," *IEEE Trans. Microw. Theory Techn.,* vol. 63, no. 7, pp. 2154-2165, Jul. 2015.

[119] V. M. R. Gongal-Reddy, S. Zhang, C. Zhang, and Q. J. Zhang, "Parallel computational approach to gradient based EM optimization of passive microwave circuits," *IEEE Trans. Microw. Theory Techn.,* vol. 64, no. 1, pp. 44-59, Jan. 2016.

[120] V. M. R. Gongal-Reddy, F. Feng, C. Zhang, S. Zhang, and Q. J. Zhang, "Parallel decomposition approach to gradient based EM optimization," *IEEE Trans. Microw. Theory Techn.,* vol. 64, no. 11, pp. 3380-3399, Nov. 2016.

[121] B. Gustavsen and A. Semlyen, "Rational approximation of frequency domain responses by vector fitting," *IEEE Trans. Power Del.*, vol. 14, no. 3, pp. 1052-1061, Jul. 1999.

[122] S. R. Schmidt and R. G. Launsby, *Understanding Industrial Designed Experiments*. Colorado Springs, CO: Air Force Academy, 1992.

[123] V. V. Prasolov, *Polynomials*, Berlin, Heidelberg: Springer-Verlag Berlin Heidelberg, 2010.

[124] H. Kabir, Y. Wang, M. Yu, and Q. J. Zhang, "High-dimensional neural-network technique and applications to microwave filter modeling," *IEEE Trans. Microw. Theory Techn.*, vol. 58, no. 1, pp. 145-156, Jan. 2010.

[125] S. Liao, H. Kabir, Y. Cao, J. Xu, Q. J. Zhang, and J. Ma, "Neural-network modeling for 3-D substructures based on spatial EM-field coupling in finite-element method," *IEEE Trans. Microw. Theory Techn.*, vol. 59, no. 1, pp. 21-38, Jan. 2011.