NOTE TO USERS

This reproduction is the best copy available.

UMI[®]

Neural Based Modeling of Nonlinear Microwave Devices and Circuits

By

Jianjun Xu, B. Eng.,

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

Ottawa-Carleton Institute for Electrical and Computer Engineering

Department of Electronics

Faculty of Engineering and Design

Carleton University

Ottawa, Ontario, K1S 5B6, Canada

© Copyright September 2004, Jianjun Xu

Library and
Archives Canada

Bibliothèque et
Archives Canada

Published Heritage
Branch

Direction du
Patrimoine de l'édition

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

ISBN: 0-612-97852-4

Our file Notre référence

ISBN: 0-612-97852-4

NOTICE:

The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

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

AVIS:

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

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

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

Abstract

Artificial Neural Networks (ANN) have been recently recognized as a useful tool for modeling and design optimization problems in RF/microwave Computer Aided Design (CAD). Neural network models can be trained from measured or simulated microwave data. The resulting neural models can be used during microwave design to provide instant answers to the task they learnt, which otherwise are computationally expensive. This thesis addresses the application of ANN to efficient and accurate modeling of nonlinear microwave devices and circuits.

Major contributions of the thesis include the adjoint neural network (ADJNN) technique, the dynamic neural network (DNN) technique and an advanced neural model extrapolation technique. The ADJNN and the DNN are two approaches that address neural based nonlinear microwave device/circuit modeling in two different cases, i.e., in the cases that the simplified topology information of such device/circuit is available or unavailable. The ADJNN approach uses a combination of circuit and neural models, where the circuit dynamics are defined by the topology and the nonlinearity is defined by ANNs. The circuit topology can be obtained from empirical models or equivalent circuits. The ADJNN technique can be used to develop a neural based model for the nonlinear device/circuit using direct current (DC) and small-signal data. The trained model can be subsequently used to predict large-signal effects in microwave circuit or system design.

The DNN approach can be used to directly model the nonlinear microwave device or circuit from its input-output data without having to rely on its internal details. The DNN model

itself can represent both dynamic effects and nonlinearity. An algorithm is developed to train the model with time or frequency domain large-signal information. Efficient representations of DNN are described for convenient incorporation of the trained model into high-level circuit or system simulation.

Further progress of neural based nonlinear microwave device/circuit modeling is made by the advanced neural model extrapolation technique. It enables neural based nonlinear device/circuit models to be robustly used in iterative computational loops, e.g., optimization and Harmonic Balance (HB), involving neural model inputs as iterative variables. Compared with standard neural based methods (i.e., without extrapolation), the proposed technique improves neural based microwave optimization and makes nonlinear circuit design significantly more robust.

The techniques developed in this thesis provide enhanced efficiency, accuracy and robustness for neural based nonlinear microwave device/circuit modeling. It is a unique contribution to further realizing the flexibility of neural based approaches in nonlinear microwave modeling, simulation and optimization.

Acknowledgements

I would like to express my sincere appreciation to my supervisor Dr. Q.J. Zhang for expert guidance, active discussions, continued assistance, encouragement, and wonderful supervision. His leadership and vision for high-level research and developmental activities has made the pursuit of this thesis a challenging, enjoyable, rewarding and stimulating experience.

I would like also to express my appreciation to Dr. Mustapha Yagoub for his contribution to my research through active involvement and invaluable collaboration. Dr. Mustapha Yagoub also provided me with direction, guidance and support during the initial stages of the thesis. I wish to thank all present colleagues in our research group, and former colleague Yonghua Fang for their nice company, productive collaboration and stimulating discussions.

Dr. Michel Nakhla and his research group are specially thanked for giving me precious academic guidance and help. Nagui Mikhail, Jacques Lemieux, and Scott Bruce are thanked for providing excellent technical support. Betty Zahalan, Peggy Piccolo and Lorena Duncan are also thanked for providing lots of help.

This thesis would not have been possible without years of support and encouragement from my parents and parents-in-law. Their guidance and love have been the precious treasures that I enjoyed through all the years of my study. Last but not the least, I would like to thank my dear wife Yang Lu. Without her support, tolerance and love, working on this thesis could have been extremely difficult.

The financial assistance provided by the Department of Electronics through a Teaching Assistantship, and the Ministry of Training, Colleges and Universities of Ontario, through an Ontario Graduate Scholarship, is gratefully acknowledged.

Dedicated to my wife

Yang Lu

for showering me with boundless love

Table of Contents

Abstract	i
Acknowledgements	iii
 CHAPTER 1	
Introduction	1
1.1 Background and Motivation	1
1.2 Contributions of the Thesis	4
1.3 Organization	7
 CHAPTER 2	
Introduction to Neural Networks and Literature Review	9
2.1 Introduction	9
2.2 Neural Based Microwave Modeling	11
2.2.1 ANN Based Microwave Modeling: Problem Statement	11
2.2.2 Neural Network Structures	13
2.2.2.1 Basic Components	13
2.2.2.2 Multilayer Perceptrons Neural Networks	14
2.2.2.3 Knowledge Based Neural Network Models	18
2.2.2.4 Radial Basis Function Networks and Wavelet Neural Networks	20
2.2.3 Neural Network Training	20
2.2.3.1 Error Derivative Computation	21

2.2.3.2 Over-Learning and Under-Learning	22
2.2.3.3 Summary of Training Process	23
2.2.4 Training Algorithms	24
2.2.4.1 Review of Back Propagation Algorithm	24
2.2.4.2 Gradient-based Optimization Methods	25
2.2.4.3 Global Training Algorithms	26
2.3 Existing Modeling Approaches for Nonlinear Microwave Devices	27
2.3.1 Physical Modeling Technique	27
2.3.2 Equivalent Circuit Modeling Technique	28
2.3.3 Neural Network Based Nonlinear Device Modeling Technique	30
2.4 Existing Modeling Approaches for Nonlinear Microwave Circuits	36
2.4.1 Behavioral Modeling Technique	36
2.4.2 Equivalent Circuit Based Approach	38
2.4.3 Model Reduction Technique	40
2.4.4 Neural Network Based Nonlinear Circuit Modeling Technique	41
2.4.4.1 Neural Network Based Behavioral Modeling Technique	41
2.4.4.2 Discrete Recurrent Neural Network Technique	45
2.5 Summary	48

CHAPTER 3

Adjoint Neural Network Technique for Microwave Modeling	49
3.1 Introduction	50
3.2 Proposed Adjoint Neural Network (ADJNN) Approach	51

3.2.2 Formulation of Two Neural Models:	
Original and Adjoint Neural Model	51
3.2.2 Basic Adjoint Neural Model Structure	55
3.2.3 Trainable Adjoint Neural Model Structure	57
3.2.4 Combined Training of the Adjoint and the Original Neural Models	58
3.2.5 Second-Order Sensitivity Analysis	68
3.3 Demonstration Examples	72
3.3.1 Example A: High-speed VLSI Interconnect Modeling and Optimization	72
3.3.2 Example B: Nonlinear Charge Modeling	79
3.3.3 Example C: Large-signal FET Modeling	81
3.5 Summary	88
 CHAPTER 4	
Dynamic Neural Network Technique for Microwave Modeling	90
4.1 Introduction	91
4.2 Dynamic Neural Network Modeling of Nonlinear Circuits:	
Formulation and Development	92
4.2.1 Original Circuit Dynamics	92
4.2.2 Formulation of Dynamic Neural Network (DNN) Model	93
4.2.3 Model Training	95
4.2.4 Use of The Trained DNN Model in Circuit Simulation	98
4.2.5 Discussions	104

CHAPTER 6

Conclusions and Future Research	144
6.1 Conclusions	144
6.2 Future Directions	146

APPENDIX A

Using Adjoint Neural Network Model and Dynamic Neural Network Model in <i>Agilent-ADS</i> for Circuit/System Simulation and Design	150
A.1 The Interface between Neural Model and ADS	151
A.2 General Function Blocks of C Code	152
A.2.1 ADJNN in ADS	154
A.2.2 DNN in ADS	154
Bibliography	156

List of Figures

Figure 2.1.	Illustration of the feedforward multilayer perceptrons (MLP) structure [1].	15
Figure 2.2.	Illustration of the structure of Knowledge based neural network (KBNN) [20]	19
Figure 2.3.	Example of a large signal equivalent circuit model of a field effect transistor [105].	29
Figure 2.4.	Incorporation of a large-signal MESFET neural network model into a harmonic balance circuit simulator [64].	31
Figure 2.5.	The Volterra-ANN device model used for modeling of nonlinear microwave device [33].	33
Figure 2.6.	The structure of the combined equivalent circuit and neural network model for nonlinear microwave devices [123].	35
Figure 2.7.	Measurable amplifier parameters for behavioral modeling [126].	37
Figure 2.8.	Behavioral model of a mixer.	38
Figure 2.9.	ANN based behavioral nonlinear microwave circuit model [33].	42
Figure 2.10.	The RNN based model structure for nonlinear microwave circuit modeling [55].	47

Figure 3.1.	A typical neuron, say the i^{th} neuron, in the original neural network.	52
Figure 3.2.	An example illustrating (a) original neural model and (b) basic adjoint neural model for sensitivity analysis.	56
Figure 3.3.	Relationship between the i^{th} original neuron and the fictitious Element Derivative Neurons (EDNs).	59
Figure 3.4.	Original neural model, adjoint neural model and EDNs. The adjoint model in this setup is trainable.	60
Figure 3.5 (a).	Illustration of the original neural model and EDNs created, for the example in Figure 3.4.	62
Figure 3.5 (b).	Illustration of EDNs and trainable adjoint model for the example in Figure 3.4.	63
Figure 3.6 (a).	Training to learn original (x, y) input-output relationship, i.e., to learn from data (x, d) .	65
Figure 3.6 (b).	Training to learn derivative information of y w.r.t x , i.e., to learn from data (x, g) .	66
Figure 3.6 (c).	Training to simultaneously learn both input-output and its derivative information, enhancing reliability of the neural model.	67
Figure 3.7 (a).	Knowledge based coupled transmission line neural model of mutual inductance (L_{12}) for VLSI interconnect optimization.	74

Figure 3.7 (b).	Basic adjoint neural model, which will be used by optimization to perform solution space analysis and synthesis of this coupled transmission line.	75
Figure 3.8.	Sensitivity verification for VLSI interconnect modeling example (a) dL_{12}/dw_1 versus s (b) dL_{12}/ds versus h .	77
Figure 3.9.	Solution space analysis: feasible regions of s - h of VLSI interconnect design for given design budgets on L_{12} .	78
Figure 3.10.	Comparison of C between the adjoint neural model and nonlinear capacitor data generated from <i>Agilent-ADS</i> .	79
Figure 3.11.	Comparison of charge model trained from nonlinear capacitance data with that from analytical integration of <i>ADS</i> capacitance formula.	80
Figure 3.12.	Large-signal FET modeling including adjoint neural networks trained by DC and bias-dependent S -parameters.	82
Figure 3.13.	Comparison between DC curves of the <i>ADS</i> Statz model (—) and our knowledge based neural FET model (o).	83
Figure 3.14.	Comparison between S -parameters of the <i>ADS</i> Statz model (—) and our knowledge based neural FET model at four of the ninety bias points : (a) $\{V_{ds} = 3.26 \text{ V}, V_{gs} = -0.6 \text{ V}\}$ and $\{V_{ds} = 0.26 \text{ V}, V_{gs} = -0.6 \text{ V}\}$, (b) $\{V_{ds} = 0.9 \text{ V}, V_{gs} = -0.6 \text{ V}\}$ and $\{V_{ds} = 0.9 \text{ V}, V_{gs} = 0.9 \text{ V}\}$.	84

$$V_{gs} = 0.0 \text{ V}\}.$$

- Figure 3.15. Complete knowledge based neural FET model, where $R_g = 4.0 \text{ } \Omega$, $R_s = 4.8994 \text{ } \Omega$, $R_d = 0.05 \text{ } \Omega$, $L_g = 0.3167 \text{ nH}$, $L_s = 0.088 \text{ nH}$, $L_d = 0.1966 \text{ nH}$, $R_x = 794.235 \text{ } \Omega$, $C_x = 20.0 \text{ pF}$, and $C_{ds} = 0.09916 \text{ pF}$ are extrinsic components. 85
- Figure 3.16. The 3-stage amplifier where the FET models used are knowledge based neural FET models trained from the proposed method following Figure 3.12. 86
- Figure 3.17. Comparison of the power amplifier large-signal responses (a) Time domain amplifier responses using ADS Statz model and our knowledge based neural FET model, (b) Output spectrum of the amplifier using ADS Statz model and our model. 87
- Figure 4.1. Schematic of Dynamic Neural Network (DNN) approach for nonlinear circuit modeling in continuous time domain. 94
- Figure 4.2. *Initial training of DNN*: to train the f_{ANN} part in time-domain using spectrum data, where $A^{(i)}$ is the time derivative operator corresponding to (4.4). 97
- Figure 4.3. Evaluation of f_{ANN} and its derivatives required during HB simulation is provided by original and adjoint neural networks, respectively. 99

Figure 4.4.	Representations of DNN for incorporation into high-level simulation. (a) Circuit representation of the DNN model. (b) HB representation of the DNN model.	101
Figure 4.5.	Amplifier circuit to be represented by a DNN model.	106
Figure 4.6.	Amplifier output: Spectrum comparison between DNN ($\boxtimes$), and ADS solution of original circuit ($\square$) at load = 50 Ω .	109
Figure 4.7 (a).	Envelope transient analysis results (output power spectrum) for DNN amplifier model with $\pi/4$ – DQPSK modulation, when the amplifier model operates at 1-dB compression point.	110
Figure 4.7 (b).	Envelope transient analysis results (output power spectrum) for DNN amplifier model with $\pi/4$ – DQPSK modulation, when the amplifier model operates at 10-dB compression point.	111
Figure 4.8.	Amplifier 2-tone simulation result from DNN, which is trained under 1-tone formulation: Spectrum comparison between DNN ($\boxtimes$) and ADS solution of original circuit ($\square$).	113
Figure 4.9.	Amplifier 2-tone simulation result from DNN: Time-domain comparison between DNN (—) and ADS solution of original circuit (o).	114
Figure 4.10.	Mixer equivalent circuit to be represented by a DNN model.	116

Figure 4.11.	Mixer V_{IF} output: Time-domain comparison between DNN (—) and ADS solution of original circuit (o).	118
Figure 4.12.	DBS receiver sub-system: (a) connected by original detailed equivalent circuit in ADS, (b) connected by our DNNs.	120
Figure 4.13 (a).	DBS system output: Comparison between system solutions using HB representation of DNN models (—), and circuit representation of DNN models (x).	121
Figure 4.13 (b).	DBS system output: Comparison between system solutions using DNN models (—), and ADS simulation of original system (o).	122
Figure 4.14.	Histogram of power gain of DBS system for 1000 Monte Carlo simulations with random input frequency and amplitude.	123
Figure 5.1.	Processing $\mathfrak{R}_t$ to obtain the effective set of base points for extrapolation, $\mathfrak{R}_b$.	132
Figure 5.2.	Flow-chart of the proposed model extrapolation.	134
Figure 5.3.	Coupled transmission lines for analysis of high speed VLSI interconnects. A neural network model is to be trained for this transmission line, and the model is to be used to demonstrate the proposed advanced neural model extrapolation technique.	136
Figure 5.4.	The optimization trajectory of design parameters w_1 and s in coupled transmission lines example.	137

coupled transmission lines example.

- Figure 5.5. Training region (-) and effective set of base points for extrapolation (o) shown in subspace of v_{in} and $v_{out}^{(1)}$, for the DNN power amplifier example. 139
- Figure 5.6. HB simulation of power amplifier: solid lines represent the training region and the circles represent the HB simulation history of $v_{out}^{(1)}$, $v_{out}^{(2)}$, and $v_{out}^{(3)}$. 140
- Figure A.1. Schematic and model parameters of two-port neural based nonlinear current source (I_{ds}) in Example C of Chapter 3, implemented in ADS using user-defined model. 151
- Figure A.2. Block diagram of the relationships between the three main functions in neural network based user-defined model and the ADS circuit simulator. 153
- Figure A.3. The evaluation and differentiation of f_{ANN} part of the DNN model is accomplished by the original neural model and its adjoint neural model respectively. 155

List of Tables

Table 3.1.	Relationship between inputs and outputs of adjoint neural model.	61
Table 3.2.	Example of sensitivity between perturbation technique and adjoint technique for the VLSI interconnet modeling example.	76
Table 4.1.	Amplifier: DNN accuracy from different training.	107
Table 4.2.	Mixer: DNN accuracy from different training.	117
Table 4.3.	DBS system component models: Testing error comparison (for spectrum data) between conventional behavioral model, static neural model, and DNNs.	126
Table 4.4.	DBS-receiver sub-system: Accuracy and computation speed comparisons between system simulation using conventional behavioral model, static neural model, DNNs, and detailed original circuit.	127
Table 5.1.	Convergence range (relative distance from solution) of non-extrapolated and extrapolated neural model for coupled transmission line example.	138
Table 5.2.	Convergence range (relative distance from solution) of non-extrapolated and extrapolated DNN model for modeling the power	141

amplifier input-output relationship.

Table 5.3.	Convergence range (relative distance from solution) of non-extrapolated and extrapolated DNN model for modeling the power amplifier two port input-output relationship.	143
------------	---	-----

List of Symbols

I	Identity matrix
$A(\omega, t)$	The coefficients of Inverse Fourier Transform
$A^{(i)}(\omega, t)$	The derivative of $A(\omega, t)$ w.r.t time t
$B(\omega, t)$	The coefficients of Fourier Transform
$\tilde{B}_i$	The i^{th} base point for model extrapolation
$\tilde{C}_{i,j}$ and $\tilde{C}_i$	The j^{th} element of vector $\tilde{C}_i$, and $\tilde{C}_i$ is the center of i^{th} subregion in the neural model input space
d	A generic N_y – vector containing response data of a given device or a circuit from simulation/measurement, used as desired outputs for neural network training
d_i	The d vector for a specific sample (i^{th} sample) of y
$e_m(w)$	Per-sample error function for m^{th} data sample (x_m, d_m)
E_1 and E_2	Initial and final training error for DNN training
E_d	Desired neural network accuracy (validation error)

E_T	Neural network training error
E_{T_e}	Neural network test error
E_v	Neural network validation error
$\tilde{E}$	The per sample training error from both the original and adjoint neural models
$\tilde{E}_a$	The per sample derivative training error for each sample data
$\tilde{E}_{a,k}$	The per sample training error between adjoint model and the sensitivity data for the k^{th} output neuron in the original model
$\tilde{E}_o$	The per sample training error from original neural model
$f_i(z, p_i)$	The processing function for generic neuron i in ADJNN
$f_{ANN}(x, w)$	Neural network model representing the relationship between x and y
g_j^k	The derivative training data for the derivative $\frac{dz_k}{dx_j}$
g	The derivative training data including various g_j^k
G_i and G	The training error at the output neuron in the adjoint model, i.e., adjoint neuron i , and G is a column vector of size N containing all G_i elements

G_{ss}	Small signal gain of a nonlinear power amplifier modeled by behavioral model
h	Update direction vector for weight parameters during neural network training
i_{RF} and i_{IF}	The currents of RF port and IF port used for DNN modeling of a mixer
i_{ANN}	Neural network based current source
I	Index sets containing indices of input neurons
I_{gd} , I_{gs} and I_{ds}	Gate-drain current, gate-source current and drain-source current of a transistor modeled by neural networks
$\tilde{I}_{i,j}$ and $\tilde{I}_i$	The j^{th} element of vector $\tilde{I}_i$, and $\tilde{I}_i$ is the center index of i^{th} subregion in the neural model input space
$\hat{I}$	Current signal in form of complex envelope signals
$\hat{I}_{in}$ and $\hat{I}_{out}$	In-phase component of the input signal and output signal
J	Jacobian matrix
K	Index sets containing indices of output neurons
K_c	Compression coefficient of a nonlinear power amplifier modeled by behavioral model

L	Total number of layers in a MLP neural network structure
n	Order of the Recurrent Neural Network or Dynamic Neural Network
N	The total number of neurons in the original neural network
N_b	The total number of base points in $\mathcal{R}_b$
N_l	Number of neurons in l^{th} layer of MLP neural network
N_{pt}	Number of ports of a device or a circuit
N_r	Total number of grid-subregions in the neural model input space
N_S	Number of states of the nonlinear circuit to be modeled by DNN
N_t	Total number of input-output sample pairs
N_u	Number of time inputs to the nonlinear circuit or the Dynamic Neural Network model or the Recurrent Neural Network model
N_w	The number of training parameters inside the neural network
N_x	The number of external inputs to the neural network
N_y	The number of outputs from the neural network except in Chapter 4. Within Chapter 4, it is used to represent the number of continuous time outputs from the nonlinear circuit or the dynamic neural network model

p_i	Vector of the training parameters in a generic neuron i . For regular neurons, such as sigmoid type of neurons, these parameters are the neuron connection weights. For knowledge neurons containing microwave empirical/equivalent model, it represents the parameters in the empirical/equivalent model
P_{in} and P_{out}	Input and output signal power of a nonlinear power amplifier
P_{sat} , P_{1dB} , and P_{DC}	Saturated power, power at 1-dB compression point, and DC power of a nonlinear power amplifier modeled by behavioral model
$\tilde{p}$	The index set of the base points closest to given input x
q_{ANN}	Neural network based charge source
Q_{gs}	Gate-source charge of a transistor modeled by neural networks
$\hat{Q}_{in}$ and $\hat{Q}_{out}$	Quadrature component of the input signal and output signal
S_{ij}	S -parameter from port j to port i of a device or a circuit
$\tilde{S}_j$	The number of intervals for model input x_j , used for model extrapolation
t	Time
T	This symbol is used within Chapter 4 to represent the time index

T_i	This symbol is used within Chapter 5 to represent the i^{th} training point
u	N_u -vector containing time domain inputs to the nonlinear circuit or the Dynamic Neural Network model or the Recurrent Neural Network model
$u^{(i)}(t)$	The i^{th} order derivatives of $u(t)$ with respect to t used in DNN modeling technique
$\bar{u}_m^{(i)}(t)$	The i^{th} order derivative training data for f_{ANN} used in DNN modeling technique
$\hat{u}$	The vector containing $u(t)$ for all the time samples t , $t \in T$, used in DNN modeling technique
$U(\omega)$	The Fourier Transform of DNN input $u(t)$
$\bar{U}(\omega)$	Training data for $U(\omega)$ in the form of input harmonic spectrums of the original nonlinear circuit, $\omega \in \Omega$, where Ω is the set of spectrum frequencies
$\hat{U}$	The vector containing $U(\omega)$ at all the spectrum components $\omega \in \Omega$, used in DNN modeling technique
v_{RF} , v_{LO} , and v_{IF}	The voltages of RF port, LO port, and IF port used for DNN modeling of a mixer
$\bar{v}$	State variables of original nonlinear circuit

$\hat{v}$	State variables of DNN
V_{gs} and V_{ds}	Gate-source and drain-source voltages of a transistor used as inputs to neural networks
$\hat{V}$	Voltage signal in form of complex envelope signals
$\tilde{V}$	This symbol is used within Chapter 5 to represent the parameters in quadratic function for model extrapolation
w	N_w -vector representing the training parameters inside the neural network also called as the weight vector
w_{i0}^l	An element of w representing the bias parameter of i^{th} neuron of l^{th} hidden layer
w_{ij}^l	An element of w representing the weight of the link between j^{th} neuron of $l-1^{th}$ layer and i^{th} neuron of l^{th} layer of MLP network
$w_{initial}$, w_{now} , and w_{next}	Weight parameters for neural network training algorithms, i.e., w , at the initial epoch, current epoch, and next epoch, respectively
Δw	Weight update vector, i.e., update of w , during neural network training
Δw_{now} and Δw_{old}	Current and previous weight update vector, i.e., current and previous Δw , during neural network training

$\tilde{W}$	This symbol is used within Chapter 5 to represent the weighting matrix containing weights between different base points for quadratic approximation
x_i and x	i^{th} external input to a neural network, and x is a N_x -vector containing the inputs, i.e., x_i , to the neural network
(x_i, d_i)	i^{th} data sample of (x, y) generated either from measurement or simulation
$\hat{x}$	N_y -vector containing the external inputs to the adjoint neural network
X_i	The external input to a generic neuron, say neuron i in the original model
y_k	k^{th} output from a neural network
$y_k(x_i, w)$	k^{th} output of the neural network model when the input presented to the network is x_i
y	N_y -vector containing the outputs from the neural network, i.e., containing y_k , except in Chapter 4. Within Chapter 4, it is used to represent continuous time domain outputs from the nonlinear circuit and the Dynamic Neural Network model
y_{RNN}	Discrete time domain outputs from the Recurrent Neural Network model

$\hat{\mathbf{y}}$	N_x -vector containing the outputs from the adjoint neural network except in Chapter 4. Within Chapter 4, it is used to represent the vector containing $y(t)$ for all the time samples t , $t \in T$
$\mathbf{y}^{(i)}(t)$	The i^{th} order derivatives of $y(t)$ with respect to t
$\bar{\mathbf{y}}_m^{(i)}(t)$	The m^{th} training data sample of $\mathbf{y}^{(i)}(t)$
Y_{ij}	Y -parameter (admittance parameter) from port j to port i of a device or a circuit
$Y(\omega)$	The Fourier Transform of DNN output $y(t)$
$\bar{Y}(\omega)$	Training data for $Y(\omega)$ in form of output harmonic spectrums of the original nonlinear circuit, $\omega \in \Omega$, where Ω is the set of spectrum frequencies
$\hat{\mathbf{Y}}$	The vector containing $Y(\omega)$ at all the spectrum components $\omega \in \Omega$, used in DNN modeling technique
z_i	The response of a generic neuron, say neuron i in the original model
z_i^l	Output of i^{th} neuron of l^{th} layer in MLP neural network, i.e., a specific z_i
$\mathbf{z}$	The vector containing the responses of all neurons, i.e., z_i , in the original neural model

$\hat{z}_j$	The symbol representing $\hat{z}_j^k$ for simplicity
$\hat{z}_j^k$	The response of the j^{th} adjoint neuron, representing the gradient of original neural model output w.r.t the local response of the j^{th} neuron in the original neural model, and k indicates an output neuron of interest for which sensitivity is to be computed
$\hat{\mathbf{z}}$	The vector containing the responses of all adjoint neurons, i.e., $\hat{z}_j^k$
$\hat{\hat{z}}_j$	The symbol representing $\hat{\hat{z}}_j^k$ for simplicity
$\hat{\hat{z}}_j^k$	The error propagation signal in adjoint neural model, and k indicates an output neuron of interest in original neural model
$\bar{\hat{z}}_n$	The symbol representing $\bar{\hat{z}}_n^k$ for simplicity
$\bar{\hat{z}}_n^k$	The new backpropagation from adjoint neural model into original neural model through EDNs, and k indicates an output neuron of interest in original neural model
$\bar{\hat{\mathbf{z}}}$	The vector containing all error propagation signals in adjoint neural model, i.e., $\hat{\hat{z}}_j^k$
η	Positive step size called learning rate for backpropagation neural network training
η^*	Optimal step size found by line search during neural network training

ξ	Momentum factor for backpropagation neural network training
ζ_i^l	Local error at the i^{th} neuron of l^{th} layer of MLP
δ	Kronecker functions
$\bar{\sigma}_j$	The new combined local gradient representing original and derivative training error backpropagated to neural j in the original neural network model
$\sigma(\cdot)$	Neuron activation function
ϕ_{in} and ϕ_{out}	Input and output signal phase of a nonlinear power amplifier
$\mathfrak{R}_b$	The total set of basis points for extrapolation
$\mathfrak{R}_t$	The training region of a neural network model

Glossary of Terms

ADJNN	Adjoint neural network
AEL	Application extension language
ANN	Artificial neural network
BP	Backpropagation
BPTT	Back Propagation Through Time
CAD	Computer aided design
CC	A constant amplitude and constant phase spectrum
CPU	Central processing unit
CR	A constant magnitude and random phase spectrum
CS	A constant magnitude and Schroeder phase spectrum
DBS	Direct Broadcast Satellite
DC	Direct Current
DDNN	Differential dynamic neural network
DF	Descriptive function

DNN	Dynamic neural network
EBP	Error backpropagation
EDN	Element Derivative Neurons
EM	Electromagnetics
HB	Harmonic balance
IDNN	Integral dynamic neural network
KBNN	Knowledge based neural networks
MLP	Multi-layer perceptrons
RBF	Radial basis function
RNN	Recurrent neural networks
RR	A random magnitude and random phase spectrum
RVTDNN	Real-valued time-delay neural network
VLSI	Very large scale integration

CHAPTER 1

Introduction

1.1 Background and Motivation

The effective use of CAD tools are important in designing RF/microwave circuits and systems with shrinking design margins and expanding system complexities. The need to reduce design iterations of such systems further demands that the tools be fast and reliable. This thesis addresses an important aspect of high-frequency CAD, namely, the modeling of nonlinear RF/microwave devices and circuits.

The motivation to pursue nonlinear device modeling comes from rapid technology advancements where new semiconductor devices constantly evolve and designers wish to accurately know how circuits containing these devices will perform. The conventional modeling techniques require human intuition and expertise to create an equivalent circuit topology and a nonlinear function for each of the nonlinear branches in the equivalent circuit, or manually modifying the existing models to match new data. Such conventional approaches are very inefficient. Methods that can automatically solve the modeling problem are much desired.

For nonlinear circuit modeling, the emphasis is on increasing computational efficiency without sacrificing too much accuracy with respect to a complete and detailed circuit description. The difficulty of simulating complex nonlinear RF/microwave circuits, at device level under large-signal conditions, often presents a significant productivity bottleneck for

design engineers. The computational complexity and computer memory demands of such circuit simulation result in a very long simulation time. These burdens become prohibitive when designing complex modules and RF/microwave sub-systems built of many of these circuits. It is often impossible to simulate such sub-systems in a nonlinear simulator at the fundamental component level. Therefore, much simplified but sufficiently accurate models for nonlinear functional blocks of the system are needed. This enables faster simulation at a higher level of abstraction while still representing accurately the effect of the nonlinear blocks in the overall system performance.

Artificial neural networks (ANN) have been recognized as a useful vehicle for RF and microwave modeling and design [1]. A neural network is a mathematical model typically consisting of a number of smooth switch functions and has the ability to learn and generalize arbitrary continuous multi-dimensional nonlinear input-output relationships [2]. ANN can be trained from measured or simulated data (samples) and subsequently used during circuit analysis and design. The models are fast and can represent the task behaviors it learnt which otherwise are computationally expensive. ANN can be more accurate than polynomial regression models, handle more dimensions than look-up table models, and allow more automation in model development than conventional modeling techniques.

Neural networks have been successfully used for modeling of linear components [1]. However, how to formulate ANN for nonlinear modeling remains an open subject until now.

The main motivation in this thesis is to efficiently and accurately model the nonlinear microwave devices and circuits, by fully exploring the potentials of neural networks.

When the simplified topology information of a nonlinear microwave device or circuit is available, we consider an attractive and efficient modeling approach combining the neural network models with the topology information. Here the circuit dynamics are defined by the topology which can be obtained from empirical models or equivalent circuits. The nonlinearity, in the form of unknown nonlinear current and charge sources, is defined by ANNs. However, we may not have detailed charge data and dynamic currents data for individual charge/current branches to train the ANNs. The combined model needs to be developed with commonly used DC and bias-dependent small-signal data, and subsequently can be used for large-signal circuit or system design. In order to perform the circuit simulation of the combined model, the derivatives of the model's outputs with respect to the model's inputs are needed for the circuit simulation matrix. This leads to the need of first order sensitivity analysis of the neural network models. For the purpose of letting the neural model learn from the small-signal information, the derivatives of its first order sensitivity with respect to its internal weights are required. This requires the derivation of second order sensitivity information.

When no simplified topology information of a nonlinear microwave device or circuit is available, a more generic and fundamental approach is to directly model the relationship between the time-domain dynamic input and output signals. Since in the time domain the outputs of the nonlinear device/circuit are not algebraic functions of the inputs, a straightforward use of the neural network model is not adequate. Expansion of the basic neural model formulation is needed in order to accommodate the dynamic nature of the problem. Furthermore, the formulation of the model also needs to be in a proper format for convenient incorporation of the trained model into high-level circuit or system simulation.

1.2 Contributions of the Thesis

The overall direction of the thesis is to develop neural based algorithms for nonlinear RF/microwave device and circuit modeling. The main objectives are, (a) To develop a neural based sensitivity analysis technique, which allows sensitivity analysis to be performed in more generic neural model structures including embedded microwave knowledge, (b) To develop a neural based nonlinear microwave device/circuit modeling technique that combines the existing circuit topology and neural network models, (c) To formulate a dynamic neural network that accommodates dynamic information in the network and can model nonlinear microwave device/circuit directly from its input-output data without having to rely on its internal details, and (d) To develop an advanced neural model extrapolation technique, which enables neural based nonlinear microwave device/circuit models to be robustly used in iterative computational loops involving neural model inputs as iterative variables. Specifically, in view of the above-mentioned objectives, the contributions of this thesis are summarized as follows.

- An adjoint neural network (ADJNN) technique [3][4] is developed for sensitivity analysis in neural based microwave modeling and design. The proposed method is applicable to generic microwave neural models including variety of knowledge based neural models embedding microwave empirical information. Through the proposed technique, efficient first- and second-order sensitivity analysis can be carried out within the microwave neural network infrastructure using neuron responses in both the original and the adjoint neural models.
- For the ADJNN method, a new formulation of simultaneous training of original and

adjoint neural models is derived allowing robust model development by learning not only the input/output behavior of the modeling problem but also its derivative data. This is very useful for analytically unified DC/small-signal/large-signal device or circuit modeling. In more detail, the ADJNN can exploit the conventional device or circuit models as topology knowledge and enhance those models through adding *trainable* nonlinear current or charge relationships to the model. Such trainable nonlinear relationships are especially beneficial when analytical formulas in the problem are unknown or available formulas are not suitable. By combining adjoint neural networks with the knowledge of existing device or circuit models, one can improve the existing models efficiently without having to go through the trial and error process typically needed during manual creation of empirical functions. The ADJNN method provides a new alternative for efficient generation of nonlinear device or circuit models for use in large-signal simulation and design.

- A neural network based modeling technique, for modeling of nonlinear microwave device or circuit, is formulated in the most ideal format, i.e., continuous time-domain dynamic system format. This format not only can best describe the fundamental essence of nonlinear behavior in theory, but also in practice is most flexible to fit most or nearly all needs of nonlinear microwave simulation, a task not yet achieved by the existing ANN-based techniques. The model, called dynamic neural network (DNN) [5][6] model, can be developed directly from input-output data without having to rely on internal details of the device or circuit. An algorithm is developed to train the model with time or frequency domain information. Efficient representations of the model are proposed for convenient incorporation of DNN into

high-level circuit simulation. Compared to existing neural based methods, the DNN retains or enhances the neural modeling speed and accuracy capabilities, and provides additional flexibility in handling diverse needs of nonlinear microwave simulation, e.g., time and frequency domain applications, single- and multi-tone simulations.

- An advanced neural model extrapolation technique [7], enabling the neural based nonlinear microwave device/circuit models to be robustly used in iterative computational loops involving neural model inputs as iterative variables, is developed. A new process is created in training to formulate a set of base points to represent a regular or irregular training region. An adaptive base point selection method is developed to identify the most significant subset of base points upon any given value of model input. Combining quadratic approximation with the information of the model at these base points including both the input/output behavior and its derivatives, this technique is able to reliably extrapolate the performance of the model from training range to a much larger region.
- An object-oriented implementation of the ADJNN, the DNN and the advanced neural model extrapolation algorithms is accomplished in C++. This computer program has been used in deriving the results in this thesis and incorporated into a trial version of the *NeuroModeler* software [8].

The ADJNN technique and the DNN technique are both applicable to nonlinear microwave device and circuit modeling. The former technique is more suitable to modeling of nonlinear microwave device, as many equivalent circuit models exist and they can be used as fast and

simplified circuit knowledge of such device. The latter is more suitable to modeling of nonlinear microwave circuits, where in most of the cases fast and simplified circuit models are usually not available.

1.3 Organization

The thesis is organized as follows.

In Chapter 2, an overview of ANN-based RF and microwave modeling including literature review is presented. Problem statement of neural based RF/microwave modeling is defined. Various aspects involved in neural network modeling are described from RF/microwave perspective. A review of different neural network structures and training algorithms is presented. An overview of existing techniques for nonlinear microwave device and circuit modeling is also conducted.

Chapter 3 presents the proposed adjoint neural network modeling technique. Concepts of original neural network model, adjoint neural network model and element derivative neurons (EDN) are introduced. Formulations of first- and second-order sensitivity analysis are presented. The advantages of this method are demonstrated through examples of high-speed VLSI interconnect modeling and optimization, nonlinear charge modeling, large-signal FET modeling, and a 3-stage power amplifier simulation utilizing the ADJNN technique.

In Chapter 4, the dynamic neural network modeling technique is introduced. Formulation and training of the DNN model are described in detail. Efficient representations of the model are presented for convenient incorporation of DNN into high-level circuit simulation. DNN

is demonstrated through examples of dynamic modeling of amplifiers, mixer and their use in a DBS receiver sub-system simulation.

The advanced neural model extrapolation technique, for improving the robustness of trained neural based nonlinear microwave device and circuit models, is presented in Chapter 5. The concept of effective base points for extrapolation is introduced. Formulations of model extrapolation are described. Advantages of this technique are demonstrated by examples of neural based design solution space analysis of coupled transmission lines and neural based behavior modeling and simulation of power amplifiers.

Finally, conclusions of the thesis are presented in Chapter 6 explaining how the thesis objectives have been successfully achieved. Recommendations for future research are also made.

CHAPTER 2

Introduction to Neural Networks and Literature Review

2.1 Introduction

Artificial neural networks are information processing systems inspired by the human brain's ability of learning and generalizing [1][2]. An ANN can be trained with given input-output information through a learning process involving storage of such information in the form of synaptic weights of the network. The fact that neural networks can learn arbitrary, continuous, multi-dimensional, and nonlinear input-output relationships from corresponding data has resulted in their successful use in diverse areas of engineering such as telecommunications [9], bio-medical [10], control engineering [11], pattern recognition [12], speech processing [13] and manufacturing [14].

In recent years, ANNs have been recognized as a useful vehicle for RF and microwave modeling and design [1]. Neural network models can be trained from measured or simulated microwave data and subsequently used during circuit analysis and design. The models are fast and can represent the task behaviors it learnt which otherwise are computationally expensive. Various types of input-output information in linear and nonlinear microwave design have been used for neural network learning, such as electromagnetics (EM) solutions versus geometrical/physical parameters [15]-[17], signal integrity solutions versus electrical parameters [18], transistor electrical versus electrical parameters [19], transistor electrical versus physical parameters [20], and more. The learning ability of neural networks is very

useful when analytical model for a new device is not available, e.g., modeling of a new transistor. Neural network can also generalize meaning that the model can respond to new data that has not been used during training. Neural models can be more accurate than polynomial regression models [21], handle more dimensions than look-up table models [22], and allow more automation in model development than conventional circuit models. Microwave researchers have demonstrated this approach in a variety of applications such as modeling and optimization of high-speed VLSI interconnects [16], bends [23][24], vias [25]-[28], CPW components [29][30], spiral inductors [31][32], microwave FETs [33]-[42], CMOS and HBTs [43]-[45], waveguides [46], laser diodes [47], filters [48]-[53], amplifiers [33][54], mixers [55], antennas [56]-[62], global modeling [63], yield optimization and circuit synthesis [64]-[66].

Neural network structures and training are two of the most important issues in applying neural networks to solve microwave problems. Theoretically, neural network models are black box models, whose accuracy depends on the data presented to it during training. A good collection of the training data, which is well distributed, sufficient, and accurately measured/simulated, is the basic requirement for obtaining an accurate model. However, training data collection/generation may be very expensive in the reality of microwave problem. There is a trade off between the amounts of training data needed for developing the neural model and the accuracy demanded by the application. Other issues affecting the accuracy of neural models are due to the fact that many microwave problems are nonlinear, non-smooth, or containing many variables. An appropriate structure would help to achieve higher model accuracy with fewer training data [67]. The size of the structure, i.e., the number of neurons, is also an important criterion in the development of a neural network.

Too small a neural network cannot learn the problem well (under-learning), but too large a size will lead to over-learning [1][2].

Neural network training is the most crucial step in the neural model development process. Training involves modification of neural network internal parameters in an orderly fashion. Such modification is carried out until the input-output relationship from corresponding training data is satisfactorily learnt. Neural network training is carried out using various optimization techniques that can be classified into gradient-based and non-gradient-based algorithms. Commonly used gradient-based algorithms in the RF/microwave CAD area include backpropagation (BP) [2][68], conjugate-gradient [69], quasi-Newton [70] and Levenberg-Marquardt [71]. Examples of non-gradient-based algorithms include simplex method [72], simulated annealing [73] and genetic algorithms [74]. Some of these training algorithms are reviewed in this chapter.

2.2 Neural Based Microwave Modeling

2.2.1 ANN Based Microwave Modeling: Problem Statement

Let x represent a N_x -vector containing parameters of a microwave device/circuit, e.g., gate length and gate width of a FET, or width and spacing of transmission lines. Let y represent a N_y -vector containing the responses of the device/circuit under consideration, e.g., drain current of a FET, or mutual inductance between transmission lines. The relationship between y and x can be highly nonlinear and multi-dimensional. The theoretical model for this relationship may not be available (e.g., a new semiconductor device), or theory may be too complicated to implement, or the theoretical model may be computationally too intensive for

online microwave design and repetitive optimization (e.g., 3D full-wave EM analysis inside a Monte Carlo statistical design loop). We aim to develop a fast and accurate neural model by teaching/training a neural network to learn the microwave problem. Let the neural network model be defined as

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

where w is an N_w -vector representing the model parameters inside the neural network also called as the weight vector [1].

The neural network can represent a specific microwave x - y relationship, only after learning the x - y relationship f_{ANN} through a process called training. As such, several (x, y) samples called training data, given by $\{(x_i, d_i), i = 1, 2, \dots, N_t\}$, need to be generated either from measurements or from simulation prior to training, where x_i and d_i are N_x - and N_y -dimensional vectors representing the i^{th} sample of x and y respectively, and N_t is the total number of input-output sample pairs. A basic description of the training objective is to determine w such that the difference between neural model outputs y and desired outputs d ,

$$E_T(w) = \frac{1}{2} \sum_{i=1}^{N_t} \sum_{k=1}^{N_y} (y_k(x_i, w) - d_{ik})^2 \quad (2.2)$$

is minimized [1][2][67]. Here d_{ik} is the k^{th} element of vector d_i , $y_k(x_i, w)$ is the k^{th} output of the neural network model when the input presented to the network is x_i , where i is the index of the training samples.

Once trained, the neural network model can be used to predict the output values given only the values of the input variables. Another stage called model test should also be performed by using an independent set of input-output samples, called testing data, to test the accuracy

of the neural network model [1]. Normally, the test data should lie within the same input range as the training data but contains input-output samples that are never seen in the training stage. The ability of neural models to predict y with x values different from that of training data is called the generalization ability [1][2]. A trained and tested neural model can then be used online during microwave design stage providing fast model evaluation replacing original slow physical/EM simulators. The benefit of the neural model approach is especially significant when the model is highly repetitively used in design process such as, optimization, Monte Carlo analysis and yield maximization.

When the outputs of the neural network are continuous functions of the inputs, the modeling problem is known as regression or function approximation, which is the most common case in microwave design area. In the next section, a detailed review of neural network structures used for this purpose is presented.

2.2.2 Neural Network Structures

2.2.2.1 Basic Components

A typical neural network structure has at least two basic components, namely, the processing elements and the interconnections between them [1]. The processing elements are called neurons and the connections between the neurons are known as links. The principal task of a neuron is to process information, and is characterized by a mathematical function called neuron activation function. Every link has a weight parameter associated with it. Each neuron receives stimulus from other neurons connected to it, processes the information, and produces an output. Neurons that receive stimuli from outside the network are called input neurons while neurons whose outputs are externally used are called output neurons. Neurons

that receive stimuli from other neurons and whose outputs are stimuli for other neurons in the network are known as hidden neurons. Different neural network structures can be constructed by using different neurons (i.e., neuron activation functions) and by connecting them differently [67].

2.2.2.2 Multilayer Perceptrons Neural Networks

A variety of neural network structures have been developed in the neural network community for microwave modeling and design. Feedforward neural network is a basic type of neural networks capable of approximating generic continuous and integrable functions. An important class of feedforward neural networks is multilayer perceptrons (MLP) [1][2]. Recently MLP neural models are widely used type of ANN structures in microwave device modeling and circuit design. Typically, the MLP neural network consists of an input layer, one or more hidden layers and an output layer, as shown in Figure 2.1.

Suppose the total number of layers is L . The input layer is layer 1, the output layer is layer L and hidden layers are 2, 3, ..., $L-1$. The input and output layers can also be denoted as hidden layer 1 and hidden layer L . Let the number of neurons in the l^{th} layer be N_l , $l = 1, 2, \dots, L$. Let w_{ij}^l represent the weight of the link between j^{th} neuron of $(l-1)^{th}$ hidden layer and i^{th} of l^{th} hidden layer, and w_{i0}^l be the bias parameter of i^{th} neuron of l^{th} hidden layer. Let x_i represent the i^{th} input parameter to the MLP. Let z_i^l be the output of i^{th} neuron of l^{th} hidden layer, which can be computed according to the standard MLP formulae as

$$z_i^l = \sigma \left(\sum_{j=1}^{N_{l-1}} w_{ij}^l \cdot z_j^{l-1} + w_{i0}^l \right), \quad i=1,2,\dots,N_l, \quad l=2,\dots,L-1 \quad (2.3)$$

$$z_i^1 = x_i, \quad i=1,2,\dots,N_x, \quad N_x=N_1 \quad (2.4)$$

Figure 2.1: Illustration of the feedforward multilayer perceptrons (MLP) structure [1]. Typically, the neural network consists of one input layer, one or more hidden layers, and one output layer.

where $\sigma(\cdot)$ is the activation function of hidden neurons. The outputs of MLP can be computed as

$$y_k = \sum_{j=1}^{N_{L-1}} w_{kj}^L \cdot z_j^{L-1} + w_{k0}^L, \quad k=1,2,\dots,N_y, \quad N_y=N_L \quad (2.5)$$

For function approximation, output neurons can be processed by linear function as shown in (2.5). The most commonly used activation function $\sigma(\cdot)$ for hidden neurons is the logistic sigmoid function given by

$$\sigma(\gamma) = \frac{1}{(1 + e^{-\gamma})} \quad (2.6)$$

which has property

$$\sigma(\gamma) \rightarrow \begin{cases} 1 & \text{as } \gamma \rightarrow +\infty \\ 0 & \text{as } \gamma \rightarrow -\infty \end{cases} \quad (2.7)$$

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

$$\sigma(\gamma) = \left(\frac{2}{\pi} \right) \arctan(\gamma) \quad (2.8)$$

and the hyperbolic tangent function,

$$\sigma(\gamma) = \frac{(e^{\gamma} - e^{-\gamma})}{(e^{\gamma} + e^{-\gamma})} \quad (2.9)$$

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

The universal approximation theorem [75] states that there always exists a three-layer MLP neural network that can approximate an arbitrary, continuous, multi-dimensional, nonlinear function to any desired accuracy. This forms a theoretical basis for employing MLP neural networks to approximate RF/microwave behaviors that can be functions of bias, geometrical,

and physical parameters. MLP neural networks are distributed models, i.e., no single neuron can produce the overall x - y relationship. For a given x , some neurons are switched on, some are off, and others are in transition. It is this combination of neuron switching states that enables the MLP to represent a given input-output mapping. During training, MLP's weight parameters capture or encode the problem information from the corresponding (x, y) training data.

The universal approximation theorem does not however specify as to what should be the size of the MLP network. The precise number of hidden neurons required for a given microwave-modeling problem remains an open question [76]. There is no clear-cut answer, however, the number of hidden neurons depends upon the degree of nonlinearity and the dimensionality of x and y (i.e., values of N_x and N_y). Highly nonlinear problems need more hidden neurons and smoother problems need fewer neurons [1]. Traditionally, either experience or trial-and-error has been used to settle on a reasonable number of hidden neurons. There has been significant research in the neural network area to determine proper network size, e.g. constructive algorithm [77], network pruning [78], regularization [79], etc. Neural networks with one or two hidden layers, i.e., three-layer or four-layer MLP are more frequently used and are usually suitable for RF/microwave application. The performance of neural network can be evaluated in terms of generalization capability and mapping capability. It is shown in [80] that three-layer MLP is preferred in function approximation where generalization capability is a major concern. Intuitively, 4-layer MLP would perform better in nonlinear problems in which localized behavioral components exist repeatedly in different regions of the problem space.

2.2.2.3 Knowledge Based Neural Networks

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

The structure of KBNN [20] is illustrated in Figure 2.2. The microwave knowledge is embedded as a part of the overall neural network internal section. There are six layers, which are not fully connected to each other, in the KBNN structure, namely input layer, knowledge layer, boundary layer, region layer, normalized region layer and output layer. The knowledge layer is the place where microwave knowledge resides, complementing the capability of learning and generalization of neural networks by providing additional information, which may not be adequately represented in a limited set of training data. The boundary layer can incorporate knowledge in the form of problem dependent boundary functions. The region layer contains neurons to construct regions from boundary neurons. The normalized region layer contains rational function-based neurons to normalize the outputs of region layer. The output layer contains second-order neuron combining knowledge neurons and normalized region neurons.

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

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

Figure 2.2: Illustration of the structure of Knowledge Based Neural Network (KBNN) [20].

The KBNN model typically includes six layers.

2.2.2.4 Radial Basis Function Networks and Wavelet Neural Networks

Feedforward neural networks which have only one hidden layer, and which use radial basis activation functions in the hidden layer, are called Radial Basis Function (RBF) networks. Radial basis functions are derived from the regularization theory in the approximation of multivariate functions [81][82]. It is demonstrated in [83][84] that RBF networks also have universal approximation ability. Universal convergence of RBF nets in function estimation and classification has also been proved [85].

The idea of combining wavelet theory with neural networks has been recently proposed [86][87][88]. Though the wavelet theory has offered efficient algorithms for various purposes, their implementation is usually limited to wavelets of small dimension. It is known that neural networks are powerful tools for handling problems of large dimension. Combining wavelets and neural networks can hopefully remedy the weakness of one with the other, resulting in networks with efficient constructive methods and capable of handling problems of moderately large dimension. This resulted in a new type of neural networks called wavelet networks, with only one hidden layer and wavelets as the hidden neuron activation functions.

2.2.3 Neural Network Training

The most important step in neural model development is neural network training. The training data consists of sample pairs, $\{(x_i, d_i), i = 1, 2, \dots, N_t\}$, where x_i and d_i are N_x - and N_y -dimensional vectors representing inputs and desired outputs of the neural network. Neural network training error [89] is defined as,

$$E_T(w) = \frac{1}{2} \sum_{i=1}^{N_t} \sum_{k=1}^{N_y} (y_k(x_i, w) - d_{ik})^2 \quad (2.10)$$

The purpose of neural network training is to adjust w such that the error function $E_T(w)$ is minimized [39]. Error $E_T(w)$ is a nonlinear function of w and iterative algorithms are often used to explore the w -space. Gradient-based iterative training techniques update w based on error $E_T(w)$ and error derivative information $\frac{\partial E_T(w)}{\partial w}$. Subsequent point in w -space denoted as w_{next} is determined by updating the current point w_{now} along a direction vector h as,

$$w_{\text{next}} = w_{\text{now}} + \eta h \quad (2.11)$$

Here, $\Delta w = \eta h$ is called the weight update and η is a positive step size called learning rate [67]. As an example, Backpropagation (BP) training algorithm updates w along the negative direction of the derivative (or gradient) of training error as

$$w_{\text{next}} = w_{\text{now}} - \eta \left. \frac{\partial E_T(w)}{\partial w} \right|_{w=w_{\text{now}}} \quad (2.12)$$

Neural network training can be categorized into sample-by-sample training and batch-mode training. In sample-by-sample training (also called online training) w is updated each time a training sample (x_i, d_i) is presented to the network. In batch-mode training (also known as offline training) w is updated after each epoch [39]. An epoch is defined as a stage of training, which involves presentation of all the training samples to the neural network once. For microwave modeling, batch-mode training is reported to be more effective [90].

2.2.3.1 Error Derivative Computation

As mentioned earlier, gradient-based training techniques need error derivative information $\frac{\partial E_T(w)}{\partial w}$. For the MLP neural network, the derivatives are computed using a standard

approach, often referred to as error backpropagation (EBP) [1], which is described in this section. A per-sample error function e_m is given by

$$e_m(\mathbf{w}) = \frac{1}{2} \sum_{k=1}^{N_y} (y_k(\mathbf{x}_m, \mathbf{w}) - d_{mk})^2 \quad (2.13)$$

for m^{th} data sample, $m = 1, 2, \dots, N_t$. Let ζ_i^L represent the error between i^{th} neural network output and i^{th} training data output, i.e.,

$$\zeta_i^L = y_i(\mathbf{x}_m, \mathbf{w}) - d_{mi} \quad (2.14)$$

This error at the output layer can be backpropagated to hidden layers as,

$$\zeta_i^l = \left(\sum_{j=1}^{N_{l+1}} \zeta_j^{l+1} w_{ji}^{l+1} \right) z_i^l (1 - z_i^l) \quad l = L-1, L-2, \dots, 3, 2 \quad (2.15)$$

where ζ_i^l represents local error at the i^{th} neuron of l^{th} layer. The derivative of the per-sample error in (2.13) with respect to a given weight parameter w_{ij}^l is given by,

$$\frac{\partial e_m(\mathbf{w})}{\partial w_{ij}^l} = \zeta_i^l z_j^{l-1} \quad l = L, L-1, \dots, 3, 2 \quad (2.16)$$

Finally, derivative of the training error in (2.10) with respect to w_{ij}^l can be computed as

$$\frac{\partial E_T(\mathbf{w})}{\partial w_{ij}^l} = \sum_{m=1}^{N_t} \frac{\partial e_m(\mathbf{w})}{\partial w_{ij}^l}. \text{ Using the EBP approach, } \frac{\partial E_T(\mathbf{w})}{\partial \mathbf{w}} \text{ can be systematically evaluated}$$

for any MLP neural network structure and can be provided to gradient-based training algorithms for the determination of the weight update $\Delta \mathbf{w}$ during training [90].

2.2.3.2 Over-Learning and Under-Learning

Validation error E_v and test error E_{T_e} can be defined in a manner similar to (2.10) using validation and test data sets. During ANN training, validation error is periodically evaluated

and training is terminated once a satisfactory E_v is reached [1]. After training, the quality of the neural model can be independently assessed by evaluating E_T [17].

Good learning of a neural network is achieved when both E_T and E_v have small values (e.g. 0.50%). An ANN exhibits *over-learning* [1], when it memorizes the training data but cannot generalize well (i.e., E_T is small but $E_v \gg E_T$). Possible reasons are too many hidden neurons, or insufficient training data. To remedy the situation, a certain number of hidden neurons can be deleted from the neural network and/or more samples can be added to the training data. An ANN exhibits *under-learning*, when it has difficulties learning the training data itself (i.e., $E_T \gg 0$). Possible reasons for under-learning are insufficient hidden neurons, or insufficient training, or training gets stuck in a local minimum. The suggested remedies are adding more hidden neurons, or continuing training, or perturbing the current solution w to escape from the local minimum and then continue training [76].

2.2.3.3 Summary of Training Process

Let E_d and max_epoch represent desired neural model accuracy (i.e., validation error) and maximum allowable number of epochs respectively, both specified by the user. Batch-mode neural network training using gradient-based training algorithms is summarized here.

Step 1: Set $epoch_number = 0$ and initialize neural network weights $w = w_{initial}$.

Step 2: Perform feedforward computation of the neural network for all the samples in validation data set and evaluate validation error E_v .

Step 3: If ($E_v \leq E_d$) or ($epoch_number \geq max_epoch$) stop training and go to step 6.

Step 4: Compute E_T and $\frac{\partial E_T}{\partial w}$ using all samples in training set simultaneously employing neural network feedforward computation and EBP.

Step 5: Find weight update Δw using a gradient-based training algorithm and update the weights as $w_{\text{next}} = w_{\text{now}} + \Delta w$. Set $\text{epoch_number} = \text{epoch_number} + 1$ and go to step 2.

Step 6: Perform feedforward computation of the neural network for all samples in test data set. Evaluate E_T and independently assess the quality of the trained neural model.

2.2.4 Training Algorithms

Each training algorithm has a scheme for updating the weights of the neural network such that the neural network converges to an acceptable solution, i.e., neural model predictions match corresponding target values. Some of the neural network training algorithms commonly used for RF and microwave modeling are reviewed in this section.

2.2.4.1 Review of Back Propagation Algorithm

Backpropagation (BP) [2] is the most popular algorithm for neural network training. BP is a stochastic algorithm based on the steepest descent principle [91], in which, weights are updated along the negative gradient direction as

$$\Delta w_{\text{now}} = w_{\text{next}} - w_{\text{now}} = -\eta \left. \frac{\partial e_m(w)}{\partial w} \right|_{w=w_{\text{now}}} \quad (2.17)$$

or

$$\Delta w_{\text{now}} = w_{\text{next}} - w_{\text{now}} = -\eta \left. \frac{\partial E_T(w)}{\partial w} \right|_{w=w_{\text{now}}} \quad (2.18)$$

Here, (2.17) is corresponding to sample-by-sample update and (2.18) is corresponding to batch-mode update. The basic BP suffers from slower convergence and possible weight oscillation. Addition of a momentum term to (2.17) and (2.18) as,

$$\Delta w_{\text{now}} = -\eta \left. \frac{\partial e_m(w)}{\partial w} \right|_{w=w_{\text{now}}} + \xi \Delta w_{\text{old}} = -\eta \left. \frac{\partial e_m(w)}{\partial w} \right|_{w=w_{\text{now}}} + \xi (w_{\text{now}} - w_{\text{old}}) \quad (2.19)$$

and

$$\Delta w_{\text{now}} = -\eta \left. \frac{\partial E_T(w)}{\partial w} \right|_{w=w_{\text{now}}} + \xi \Delta w_{\text{old}} = -\eta \left. \frac{\partial E_T(w)}{\partial w} \right|_{w=w_{\text{now}}} + \xi (w_{\text{now}} - w_{\text{old}}) \quad (2.20)$$

reduces the weight oscillation [68], where ξ is called the momentum factor which controls the influence of the last weight update direction on the current weight update, and w_{old} represents the last point of w .

2.2.4.3 Gradient-based Optimization Methods

The Backpropagation algorithm, which based on steepest descent principle, is relatively easy to implement. However, the error surface of neural network training usually contains planes with a gentle slope due to the squashing functions commonly used in neural networks. The error gradient values are too small for weight to move rapidly on these planes, the rate of convergence is slow. The rate of convergence could be very slow when the steepest decent method encounters “narrow valley” in the error surface where the direction of gradient is close to the perpendicular direction of the valley. The update direction oscillates back and forth along the local gradient.

Gradient-based optimization techniques, which determine the update direction using derivative information of $E_T(w)$ can help to improve the rate of convergence [92]-[94]. Let

h be the direction vector, η be the learning rate, w_{now} be the current value of w , then the optimization will update w such that

$$E_T(w_{\text{next}}) = E_T(w_{\text{now}} + \eta h) < E_T(w_{\text{now}}) \quad (2.21)$$

The principal difference between various descent algorithms lies in the procedure to determine successive update directions (h) [95]. Once the update direction is determined, the optimal step size could be found by line search along h ,

$$\eta^* = \min_{\eta > 0} E_T(\eta) \quad (2.22)$$

where

$$E_T(\eta) = E_T(w_{\text{now}} + \eta h) \quad (2.23)$$

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

Commonly used gradient-based algorithms in the RF/microwave CAD area include, conjugate-gradient [69], quasi-Newton [70] and Levenberg-Marquardt [71].

2.2.4.3 Global Training Algorithms

Another important class of methods uses random optimization techniques that are characterized by a random search element in the training process allowing the algorithms to escape from local minima and converge to the global minimum of the objective function. Examples include simulated annealing [73] which allows the optimization process to jump out of a local minimum through an annealing process controlled by a temperature parameter,

and genetic algorithms [74] that evolve the structure and weights of the neural network through generations in a manner similar to biological evolution.

Since the convergence of pure random search techniques tends to be very slow, a more viable method is the hybrid method, which combines both conventional gradient-based training algorithms and random optimization concepts, e.g. [96]. During the training with conjugate gradient method, if a flat error surface is encountered, the training algorithm switches to the random optimization method. After the training escapes from the flat error surface, it switches back to the conjugate gradient method.

We have discussed fundamental issues about the neural networks in structure, activation function, and model training. In next step, we will present some existing modeling methods for nonlinear microwave devices and circuits.

2.3 Existing Modeling Approaches for Nonlinear Microwave Devices

2.3.1 Physical Modeling Technique

The classical approach to obtain a suitable compact device model for circuit simulation has been to make use of available physical knowledge, and to forward that knowledge into a numerically well-behaved model [97]-[104]. Two important types of physical models that are applied to device design and characterization are described in [97]. The most straightforward of these is based on a derivative of equivalent circuit models, where the circuit element values are quantitatively related to the device geometry, material structure,

and physical processes. The second approach is more fundamental in nature and is based on the rigorous solution of the carrier transport equations over a representative geometrical domain of the device. These models use numerical solution schemes to solve the carrier transport equations in semiconductors often accounting for hot electrons, quantum mechanics, EM, and thermal interaction. In particular, a key advantage is that physical models allow the performance of the device to be closely related to the fabrication process, material properties, and device geometry. This allows performance, yield, and parameter spreads to be evaluated prior to fabrication, resulting in a significant reduction in the design cycle. Furthermore, since physical models can be embedded in circuit simulations, the impact of device–circuit interaction can be fully evaluated. A further advantage of physical models is that they are generally intrinsically capable of large-signal simulation. On the other hand, a major disadvantage of physical modeling is that it usually takes a long time to develop a good model for a new device. That has been one of the major reasons to explore alternative modeling techniques.

2.3.2 Equivalent Circuit Modeling Technique

One commonly used modeling approach for microwave device is the lumped equivalent circuit technique [1][105]–[118], where the equivalent electrical circuit contains nonlinear controlled voltage or current sources, together with (linear or nonlinear) parasitic resistors, inductors and capacitors. All nonlinear elements are represented by empirical functions containing several so-called “model parameters”. Dedicated procedures allow to extract the value of these parameters out of DC and small signal *S*-parameter measurements. As an example, an equivalent circuit model of a field effect transistor [105] is shown in Figure 2.3.

Compared to detailed physical models, the equivalent circuit model is much faster, but only accurate in specific cases. Developing such models requires experience and involves a trial-and-error process to find appropriate circuit topology and the values of the circuit elements. Moreover, an equivalent circuit model may not have direct links with the physical/process parameters of the device. Empirical formulas for such links may exist, but the accuracy cannot be promised when applied to different devices.

D_f , D_r , C_{gs} and i_d are nonlinear elements. For example [1],

$$i_d = I_{dss} \left(1 - \frac{V_g}{V_p}\right)^2 \tanh\left(\frac{a' V_d}{V_g - V_p}\right)$$

$$V_p = V_{po} + \gamma V_d$$

where I_{dss} , a' , V_{po} , and γ are parameters of the nonlinear element

Figure 2.3. Example of a large signal equivalent circuit model of a field effect transistor [105].

2.3.3 Neural Network Based Nonlinear Device Modeling Technique

Recently neural networks are used for nonlinear microwave device modeling to meet the requirement for fast and accurate model development. Several modeling methods have been published [20][33][36][41][44][64][119][120][122][123][124].

The direct modeling approach, in which the component external behaviors are directly modeled by neural networks, has been used in transistor modeling. It has been applied to model DC characteristics of a physics-based MESFET [20], small-signal HBT device [44] and large-signal MESFET device [33][64][119][120]. As an example, in [64] a straightforward formulation of large signal models to describe terminal currents and charges of nonlinear devices as nonlinear functions of the device parameters and the bias conditions is described. In this example, the terminal currents and charges for different configurations of MESFET were simulated at a number of bias points using *OSA90* [121] with the *Khatibzadeh* and *Trew* model [102]. The neural network model has six inputs namely gate-length, gate-width, channel thickness, doping density, gate voltage and drain voltage. The terminal currents and charges at drain, gate and source electrodes are the model outputs, leading to a total of six output parameters. Since the neural model directly describes terminal currents and charges as nonlinear functions of device parameters, it can be conveniently used in harmonic balance environment. The trained large-signal neural models were plugged into a circuit simulator as shown in Figure 2.4. The large-signal MESFET neural model was then used to satisfactorily perform DC, small-signal and HB simulations. The work of [120] has successfully demonstrated this technique to modeling of HEMTs and nMOS devices, using full two-port vectorial large-signal measurements as training data.

Figure 2.4. Incorporation of a large-signal MESFET neural network model into a harmonic balance circuit simulator [64]. The ANN model can be included as an additional nonlinear subnetwork.

As another example, a neural based device modeling technique based on time varying Volterra series has been developed in [33]. In this example, the relationships between terminal currents and voltages of a nonlinear device are related by the time varying Volterra series,

$$i_k(t) = I_{k0}(v_1(t), \dots, v_{N_{pt}}(t)) + \sum_{j=1}^{N_{pt}} \sum_{m=-F}^F Y_{kj}(v_1(t), \dots, v_{N_{pt}}(t), \omega_m) V_{jm} e^{j\omega_m t} \quad (2.24)$$

where $k=1, \dots, N_{pt}$, and N_{pt} is the number of ports. The DC term I_{k0} is defined by DC device measurements, and the time varying kernel Y_{kj} is directly related to the measured device bias dependent Y -parameters, on all the frequency ranges $[\omega_{-F}, \omega_F]$. In this example, neural networks were used to model I_{k0} as a function of device bias, and bias dependent Y -parameters as a function of device bias and ω . The training data is directly obtained from automatic measurement setup. The Volterra-ANN device model is shown in Figure 2.5. This technique has been successfully demonstrated in [33] for modeling of nonlinear microwave transistor that is further used in microwave amplifier.

An indirect modeling approach combines known equivalent circuit models together with neural network models to develop more efficient and flexible models. As described in Section 2.3.2, the lumped equivalent circuit approach is a traditional approach to transistor modeling. Developing such models requires experience and involves a trial-and-error process to determine a matching topology. Moreover, equivalent circuit parameters may not be related to the physical/geometrical parameters of the device under consideration. Empirical formulae for such relations exist and neural networks can easily learn these relationships. A hybrid approach that utilizes existing knowledge in the form of known equivalent circuit and empirical formulae, together with the powerful learning and

generalization abilities of neural networks was demonstrated for modeling large-signal behaviors of MESFET [41] and HEMT [122]-[124].

Figure 2.5. The Volterra-ANN device model used for modeling of a nonlinear microwave device [33].

As an example, a neural based nonlinear device modeling method is described in [123] for large-signal modeling of a HEMT. For future implementation in commercially available simulators, the bias-dependent behavior of the HEMT was represented in terms of conventional small-signal equivalent circuit elements, i.e., the bias-dependent intrinsic C_{gs} , R_i , C_{gd} , g_m , d_{τ} , g_{ds} , and C_{ds} . The neural networks are used to model the nonlinear relationships between those intrinsic elements and the terminal voltages (V_{gs} and V_{ds}). After training with measurement data, the complete neural based nonlinear device model, which combines the equivalent circuit and neural network models as shown in Figure 2.6, is able to be used in circuit simulator for high-level circuit design. In [124], a dynamically configurable combination of empirical equations and neural networks is developed to increase the flexibility of a nonlinear device model's capabilities. The framework for this model is a common-source large-signal equivalent FET circuit. With the exception of the drain current source, all of the nonlinear elements of the circuit are configurable to either empirical or bias-dependent neural network controlled components, which gives the modeler freedom to tailor the model to different applications without having to redevelop it. The neural network architecture employed is based on the knowledge base algorithm, that is implemented into device models to increase simulation accuracy while reducing and simplifying development.

As the application of neural network modeling technique proved, trained neural models with measurement data can represent DC, small-signal and large-signal behaviors of a new device, even if the device theory/equations are still unavailable. Because neural network can learn the nonlinearity much more automatically and efficiently than manually formulating a nonlinear function, it is a very suitable and efficient alternative for such modeling activities.

Equivalent circuit-Neural network model

Figure 2.6. The structure of the combined equivalent circuit and neural network model for nonlinear microwave devices [123].

2.4 Existing Modeling Approaches for Nonlinear Microwave Circuits

2.4.1 Behavioral Modeling Technique

A popular nonlinear microwave circuit modeling approach is the behavioral modeling technique [125][126]. In this approach, the input/output behaviors of the nonlinear circuits are characterized by a set of well-defined parameters. When new input signals are fed, the output signal then can be calculated from these parameters and inputs.

A simple version of this approach uses a set of simple parameters to describe different aspects of the relationship between circuit input and output signals. For example, as illustrated in Figure 2.7, a nonlinear class-A power amplifier can be modeled with following parameters, small signal gain G_{ss} , compression coefficient K_c , saturated power P_{sat} , power at 1-dB compression point P_{1dB} , 3rd order intercept point $IP3$, third-order intermodulation $IM3$, DC power P_{DC} , power added efficiency PAE , and phase distortion $AM-PM$ [126]. The task of behavioral modeling technique is to formulate the static mapping between these parameters and the input signals, namely, the DC bias, the input power P_{in} , the frequency f , and the phase ϕ_{in} . It can be accomplished by various curve-fitting techniques, such as linear regression, look-up table (linear interpolation), logarithmic regression, power function regression, exponential regression, and spline curve fitting, etc. It is worthy to mention that feedforward neural network, e.g., MLP, can also easily do the function approximation task.

In a more systematic and generic behavioral modeling approach, the models are built in frequency-domain [127]-[133].

Figure 2.7. Measurable amplifier parameters for behavioral modeling [126].

In [127], the spectral components of input and output signals at the ports in the circuit are used to build the model. The task of this modeling is to generate functions to map the nonlinear relationship between all the input spectral components and all the output spectral components. As an illustration, behavioral model for a mixer is shown in Figure 2.8. Though from a mathematical point of view, the modeling procedure is a multi-dimensional function approximation that starts from measurement or simulation data. Special attention should be paid to the huge input space, which includes all the spectral components of all the input ports. It is pointed out in [127] that several concepts are needed to make this approach successful in practice. The first one is the time-invariance concept, which means applying a frequency proportional phase shift to input spectral components will result in the same frequency proportional phase shift to all output spectral components [133]. The second one

is applying the linearization concept to those relatively small spectral components so that the superposition principle can be applied [134]. This concept reduces the dimensionality of the input space to a manageable size. But even with these concepts, the function-approximation is not a small issue due to the high nonlinearity of the relationship.

In [128], a technique referred as data-based behavioral modeling technique is developed. After initial extraction of data for the circuit level devices and subsequent generation of their behavioral models, such models, together with data file used to generate the model, can be used as building blocks for RF/microwave receivers and transmitters and allow a fast but accurate system level simulation that cannot be completed at the circuit level.

Figure 2.8. Behavioral model of a mixer.

2.4.2 Equivalent Circuit Based Approach

Another popularly used microwave circuit modeling approach is the equivalent circuit based approach. Typically, the techniques of this approach result in a simpler circuit with lumped nonlinear components compared to the original nonlinear microwave circuit. Several techniques based on equivalent circuit for large-scale nonlinear microwave circuits have been proposed. Early in 1974, techniques known as “circuit simplification” and “circuit

build-up” were used to generate models for operational amplifier [135]. Based on the understanding and experience, different parts of the original nonlinear circuit are either simplified by using smaller number of the same ideal elements or re-built by generating a new circuit configuration. The parameters and values of elements are determined by matching certain external specification of the original circuit. Later, an automated modeling algorithm for analog circuits was introduced in [136]. It can be applied to general nonlinear microwave circuits if the original circuit satisfies following conditions:

- All the components in the circuit can be modeled as independent current sources, resistors, capacitors, and voltage-controlled current sources.
- Resistors, capacitors, and controlled sources are not required to be linear, i.e., they can be described by branch equations of the form $i_d = f_d(v_d)$ or $q_d = f_d(v_d)$, where i_d is the current flowing through the device, q_d is the charge on the capacitor, f_d is a function depending on the device and v_d is the controlling branch voltage. Furthermore, it is reasonably assumed that there exists a constant $c_{min} > 0$ such that $(dq_d/dv_d) \geq c_{min}$ for all voltages v_d and all capacitors in the circuit.
- There are capacitors connecting the ground node to all other nodes of the circuit.

Besides the assumptions, a template, e.g., the topology of the equivalent circuit needs to be supplied. With the provided template, the dynamics of the model can be formulated in time-domain using general q - v - i form with outputs v^0 as the solution:

$$\begin{cases} \dot{q} = -\phi(v) + Pi \\ v^0 = \psi(q) \end{cases} \quad (2.25)$$

where φ and ψ are nonlinear functions, P is matrix with 1 for nodes connected with current source i and 0 elsewhere.

The algorithm then determines the value of parameters in the equivalent circuit by minimizing the difference of solution v^0 between the original circuit and the model under the same excitation i . Techniques used in optimal control and nonlinear programming are employed to do the minimization procedure. The resulting model can be used for general-purpose circuit simulation.

Though with the merits of automation and generating general-purpose models, this technique has several disadvantages. Firstly, the requirements of the algorithm restrict the range of its application. Secondly, providing the template equivalent circuit of the model is not a trivial task which requires the good understanding of the original circuit and practical experience. Usually, a trial and error procedure is needed to get an appropriate equivalent circuit for a large-scale nonlinear circuit.

2.4.3 Model Reduction Technique

When the full equations of the original nonlinear circuits are available and accessible, a technique based on Krylov-subspace is proposed in [137][138]. This technique can reduce the order of the original system to a user specified number $\tilde{q}$ so that the first ' $\tilde{q}$ ' derivatives of the time-response of the original system are retained.

In this technique, the nonlinear state-based dynamic equations of the original large-scaled nonlinear circuit are first formulated in time domain. Taylor expansion is then applied to time-related parameters in both sides of these equations with respect to time, which

including system states and input signals. The Krylov subspace of the original nonlinear system is formulated as a matrix $\bar{K}$ by putting together the first $\tilde{q}$ Taylor coefficients of all the system states. The reduced system is composed by a set of new states, which is the result of performing a congruent transformation to the original system states using Q matrix resulted from QR decomposition on the matrix $\bar{K}$. The new system has an order of $\tilde{q}$ and they are theoretically identical to the first $\tilde{q}$ orders of the original nonlinear system. By replacing the original nonlinear circuit with the new system in circuit simulation, a significant speed improvement is found.

2.4.4 Neural Network Based Nonlinear Circuit Modeling Technique

2.4.4.1 Neural Network Based Behavioral Modeling Technique

Behavioral modeling techniques normally require a computationally efficient approximation of complex multivariable input/output relationships, which may be effectively handled by artificial neural networks. Recently several ANN based behavioral modeling techniques [1][33][127][139][140][141] have been proposed for modeling of nonlinear RF/microwave circuits. These works demonstrated neural networks as a useful alternative to the conventional modeling approaches.

For example, in [33] a neural network based behavioral modeling technique is developed, which considers a new circuit model expressing input-output relations in the following way,

$$\begin{aligned}\hat{I}_1(t) &= f_1(\hat{V}_1(t), \hat{V}_2(t)) \\ \hat{I}_2(t) &= f_2(\hat{V}_1(t), \hat{V}_2(t))\end{aligned}\tag{2.26}$$

where $\hat{I}$, $\hat{V}$ are complex input-output envelope signals, formulated from the input-output time domain signals of the nonlinear microwave circuit, i.e., $i(t)$, $v(t)$ as,

$$\begin{aligned}
i_1(t) &= \hat{I}_1(t) \cdot e^{j\omega_0 t} \\
i_2(t) &= \hat{I}_2(t) \cdot e^{j\omega_0 t} \\
v_1(t) &= \hat{V}_1(t) \cdot e^{j\omega_0 t} \\
v_2(t) &= \hat{V}_2(t) \cdot e^{j\omega_0 t}
\end{aligned} \tag{2.27}$$

and ω_0 is the carrier frequency. In this example, neural networks are used to formulate the nonlinear functions f_1 and f_2 . The overall neural network based behavioral model is shown in Figure 2.9. This technique has been successfully demonstrated in [33] for modeling of nonlinear microwave amplifier directly from measurements.

Figure 2.9. ANN based behavioral nonlinear microwave circuit model [33].

As another example, in [139], a bidirectional ANN based behavioral modeling technique is developed. The model is equally applicable to nonlinear microwave circuit with and without frequency conversion between the input and the output ports. Linear distortion and small-

signal dynamics are accounted for by the model through the frequency-dependent two-port conversion admittance matrix, reducing to the ordinary admittance matrix in the absence of frequency conversion. Nonlinear distortion and large-signal dynamics are introduced by a couple of frequency-dependent descriptive functions (DF) relating the linear and nonlinear responses (i/o currents).

For example, in the case of modeling nonlinear microwave circuit without frequency conversion between the input and the output ports, the DF for port 1 and port 2 are

formulated as $\frac{F_1(A_1, A_2, \varphi; \omega_0)}{Y_{11}(\omega_0)A_1 + Y_{12}(\omega_0)A_2 \exp(j\varphi)}$ and $\frac{F_2(A_1, A_2, \varphi; \omega_0)}{Y_{21}(\omega_0)A_1 + Y_{22}(\omega_0)A_2 \exp(j\varphi)}$, respectively.

Here $F_1, F_2, A_1, A_2, \varphi$, and ω_0 consist of the complete definitions of phasors and frequencies for port voltages and port currents, i.e.,

$$V_1 = A_1 \quad (2.28)$$

$$V_2 = A_2 \exp(j\varphi)$$

$$I_1 = F_1(A_1, A_2, \varphi; \omega_0)$$

$$I_2 = F_2(A_1, A_2, \varphi; \omega_0)$$

and Y_{11}, Y_{12}, Y_{21} , and Y_{22} forms the two-port admittance matrix. The DF are computed by the harmonic balance simulation and are efficiently approximated by neural networks, which have three or four inputs (A_1, A_2, φ , and possibly ω_0) and two outputs (the real and imaginary parts of DF for port 1 or port 2).

The resulting ANN based behavioral model is fast to compute, can accurately handle broadband modulated signals, and is fully bidirectional. This means that the model can be routinely used to analyze high-level systems where bidirectional signal flow takes place, e.g.,

due to mismatches between the interconnected subsystems, or even due to the presence of filters operating in the stopband. The excellent performance of this behavioral model has been demonstrated by harmonic balance analysis, in the cases of both isolated and interconnected subsystems, under both sinusoidal and modulated-RF drive in [139].

As another recent neural based behavioral modeling technique, the real-valued time-delay neural network (RVTDNN) [140] is developed for dynamic modeling of the base band nonlinear behaviors of third-generation (3G) base-station power amplifiers.

The RVTDNN model utilizes the two components of the input signal (in-phase $\hat{I}_{in}$, quadrature $\hat{Q}_{in}$) to predict the correspondent two components (in-phase $\hat{I}_{out}$, quadrature $\hat{Q}_{out}$) of the output signal. Considering the memory effects, the base band output $\hat{I}_{out}$ and $\hat{Q}_{out}$ components of the power amplifier at instant k are a function of $\hat{p}$ past value of the base band input $\hat{I}_{in}$ and $\hat{q}$ past values of the base band input $\hat{Q}_{in}$, as follows:

$$\begin{aligned}\hat{I}_{out}(k) &= f_I(\hat{I}_{in}(k), \hat{I}_{in}(k-1), \dots, \hat{I}_{in}(k-\hat{p}), \hat{Q}_{in}(k), \hat{Q}_{in}(k-1), \dots, \hat{Q}_{in}(k-\hat{q})) \\ \hat{Q}_{out}(k) &= f_Q(\hat{I}_{in}(k), \hat{I}_{in}(k-1), \dots, \hat{I}_{in}(k-\hat{p}), \hat{Q}_{in}(k), \hat{Q}_{in}(k-1), \dots, \hat{Q}_{in}(k-\hat{q}))\end{aligned}\quad (2.29)$$

where neural networks are used to formulate the nonlinear functions f_I and f_Q .

Time- and frequency-domain simulation of a 90-W LDMOS power amplifier using this neural based model exhibit a good agreement between the RVTDNN behavioral model's predicted results and measured ones along with a good generality [140]. Moreover, dynamic AM/AM and AM/PM characteristics obtained using this model demonstrated that the RVTDNN can track and account for the memory effects of the power amplifiers well [140].

This RVTDDN model requires a significantly reduced complexity and shorter processing time in the analysis and training procedures, when driven with complex modulated and highly varying envelope signals, than previously published neural based power amplifier models.

Addressing another perspective of neural based nonlinear circuit behavioral modeling, the work of [141] focuses on the use of multisines in the experiment design. More specifically, it evaluates four types of multisine excitations with respect to their ability to generate accurate behavioral models. The four types of multisine excitations are: a multisine with a constant amplitude and constant phase spectrum (abbreviated by CC), a constant magnitude and random phase spectrum (CR), a constant magnitude and Schroeder phase spectrum (CS), and finally a random magnitude and random phase spectrum (RR). The evaluation is carried out by using a time domain neural based behavioral model described in [142]. Based on the experimental results, it is concluded in [141] that the RF trajectories of the multisine with a constant amplitude and random phase spectrum (CR) are the most uniformly spread, and so this multisine type of excitation is more appropriate for obtaining accurate behavioral models.

2.4.4.2 Discrete Recurrent Neural Network Technique

Feedforward neural networks are well known for their ability to map static input-output relationships accurately. To model nonlinear circuit responses in time-domain, a neural network that can include temporal information is necessary. Recurrent neural networks (RNN) [55] were found to be one of the suitable techniques for this purpose.

The structure of a typical RNN is shown in Figure 2.10. The inputs of the recurrent neural

network are time-varying inputs u . The outputs of the recurrent neural network are denoted by vector y_{RNN} . The first hidden layer of RNN contains buffered (time-delayed) history of y_{RNN} fed back from the output layer, and buffered history of u . The second hidden layer contains sigmoid neurons. The part in the model structure from input layer, hidden layer, to output layer is a feed forward neural network denoted as $f_{ANN}(x, w)$, where w is a vector containing all the connection weights in the feed forward neural network. The overall neural network realizes a nonlinear relationship:

$$y_{RNN}(k) = f_{ANN}(y_{RNN}(k-1), \dots, y_{RNN}(k-n), u(k-1), \dots, u(k-n), w) \quad (2.30)$$

where $y_{RNN}(k)$ and $u(k)$ are simplified notations for $y_{RNN}(k\tau)$ and $u(k\tau)$ respectively, τ is the time sampling interval, and the number of delay buffers n is the order of dynamics in the RNN model, which represent effective order of original nonlinear circuit as seen from input-output data.

The RNN can be trained to learn the dynamic characteristics of a nonlinear RF/microwave circuit. For such use, the training data can be a set of input and output waveforms of the nonlinear circuit under consideration, which can be obtained from measurements or simulation. Since the present outputs of the neural model not only depend upon the present inputs, but also on the previous inputs and outputs, a novel BP training scheme called backpropagation-through-time (BPTT) needs to be used [143]. Once being trained, the RNN macromodel provides fast prediction of the full analog behavior of the original circuit, which can be useful for high-level simulation and optimization.

Figure 2.10. The RNN based model structure for nonlinear microwave circuit modeling [55].

2.5 Summary

In this chapter, an overview of RF/microwave computer-aided design approach based on artificial neural networks has been presented. Fundamental concepts such as problem statement of ANN-modeling, neural network structures, and training algorithms, have been systematically described.

Existing modeling techniques for nonlinear microwave device and circuit have been reviewed. In each category, neural network based methods have been developed, which demonstrated neural networks as a useful alternative to the conventional approaches. Neural based models have been used for modeling of DC, small-signal and large-signal behaviors of nonlinear microwave device/circuit. The models can be realized in frequency-domain or time-domain formats, and can be a pure neural network, or a knowledge based neural network in which RF/microwave information is utilized together with neural networks.

Research and development efforts are further required to extract the full potential of neural networks to formulate neural models that can represent nonlinear microwave devices or circuits more efficiently and more accurately. This chapter gives the basic foundation of the state-of-the-art in this area and helps us better understanding of the thesis contributions presented in the following chapters.

CHAPTER 3

Adjoint Neural Network Technique for Microwave Modeling

In this chapter, one of the major contributions of this thesis work, namely, the Adjoint Neural Network (ADJNN) technique [3][4] is presented. The proposed ADJNN technique can be used to develop a combined model of circuit and neural networks for the nonlinear device/circuit using DC and small-signal data. The trained model can be subsequently used to predict large-signal effects in microwave circuit or system design.

The ADJNN technique aims to address several practical challenges in RF and microwave modeling and design, e.g., neural based sensitivity analysis and efficient/accurate nonlinear microwave device/circuit modeling. The proposed neural based sensitivity analysis is applicable to generic microwave neural models including variety of knowledge based neural models embedding microwave empirical information. Through the proposed technique, efficient first- and second-order sensitivity analysis can be carried out within the microwave neural network infrastructure using neuron responses in both the original and the adjoint neural models. A new formulation of simultaneous training of original and adjoint neural models allows robust model development by learning not only the input/output behavior of the modeling problem but also its derivative data. This feature allows the neural based nonlinear microwave device/circuit model, which is a combination of circuit and neural models, to be developed with DC/small-signal data. The trained model can be subsequently used to predict time and frequency domain large-signal effects in high-level circuit or system design.

3.1 Introduction

In recent years, neural network has been recognized as a useful vehicle for RF and microwave modeling and design. Research work presented in this chapter addresses a new task in this area, that is, neural based sensitivity analysis. Sensitivity information is very important for circuit optimization [144][145], and for unified DC/small-signal/large-signal modeling and circuit design [146]. In the case of neural networks, first-order sensitivity analysis has been studied, for example, for networks with binary responses for signal processing purposes [2], and for multilayer perceptron structures used in microwave modeling and design [147][148]. However, to perform sensitivity analysis in more generic neural model structures including embedded microwave knowledges, and to train the networks to learn from sensitivity data that arise during microwave modeling remain an unsolved task.

For the first time, a novel adjoint neural network (ADJNN) sensitivity analysis technique is presented, which allows exact sensitivity to be calculated in a general neural model accommodating microwave empirical functions, equivalent circuit as well as conventional switch type neurons in an arbitrary neural network structure. The adjoint neural network structure is excited by a unit excitation corresponding to the output neurons in the original neural network. A new formulation allows the training of the adjoint neural models to learn from derivative training data. An elegant derivation is presented where the first- and second-order derivative calculation are carried out using the neural network infrastructure through a combination of backpropagation processes in both the original and adjoint neural networks. Using the second-order derivative, we are able to train a neural network model to learn not only microwave input/output data but also its derivative information, which is very useful in simultaneous DC/small-signal/large-signal device or circuit modeling.

3.2 Proposed Adjoint Neural Network (ADJNN) Approach

3.2.1. Formulation of Two Neural Models: Original and Adjoint Neural Model

Two models, one called the original neural network model, and the other defined as the adjoint neural network model, are utilized in the proposed sensitivity analysis technique. Each model consists of neurons and connections between neurons. Each neuron receives and processes stimuli (inputs) from other neurons and/or external inputs, and produces a response (output). Here we introduce a generic framework in which microwave empirical and equivalent models can be coherently represented in the neural network structures, and connections between neurons can be arbitrary allowing different types of microwave neural structures to be included.

Suppose for a generic neuron, say neuron i in the original model, the response is z_i and the external input to this neuron is X_i . Let N be the total number of neurons in the original neural network and $\mathbf{z} = [z_1, z_2, \dots, z_N]^T$. In order to accommodate microwave empirical knowledge, we use a notation $f_i(\mathbf{z}, \mathbf{p}_i)$ to represent the processing function for neuron i where $\mathbf{p}_i$ could represent either the neuron connection weights or the parameters of a microwave empirical/equivalent model. See Figure 3.1.

The collection of $\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_N$ forms the weight vector $\mathbf{w}$ for the overall neural model. For

example, if neuron i is a sigmoid switch neuron, then $f_i(\mathbf{z}, \mathbf{p}_i) = \frac{1}{1 + e^{-\mathbf{p}_i^T \cdot \mathbf{z}}}$, where $\mathbf{p}_i$ is a N -

vector and its elements represent connection weights between neuron i and other neurons.

Figure 3.1. A typical neuron, say the i^{th} neuron, in the original neural network. The neuron receives stimulus from responses of other neurons $z_j, j < i$, processes the stimulus using a processing function $f_i(z, p_i)$ and produces a response z_i . p_i is a vector of parameters for the processing function. X_i is an external stimulus.

For another example, $f_i(z, p_i)$ could represent an empirical formula for FET drain current versus terminal voltages and physical/geometrical parameters [20]. If df_i/dz_j is non-zero (or zero), then neuron i is (or is not) connected from neuron j . In such way, this formulation allows us to represent not only multilayer perceptrons, but also arbitrary connections between neurons, and knowledge based neural networks.

A neuron which receives a stimulus from outside the neural network is called an input neuron. A neuron whose response becomes the output of the overall neural network model is called an output neuron. A neuron whose stimulus is from responses of other neurons, and whose response becomes stimulus to other neurons is called a hidden neuron. Let I and K be defined as index sets containing indices of input neurons and output neurons, respectively, i.e.,

$$I = \{i \mid \text{if stimulus to neuron } i \text{ is from neural model external inputs, i.e., } x([x_1, x_2, \dots, x_{N_x}])\}$$

$$K = \{k \mid \text{if response of neuron } k \text{ is an output of the overall neural model, i.e., } y([y_1, y_2, \dots, y_{N_y}])\}$$

where N_x and N_y are the input and output number of the neural model, respectively.

Assuming the neuron indices are numbered consecutively starting from the input neurons, through hidden neurons to the outputs neurons. The feedforward calculation of the original model can be defined as

$$z_i = f_i(z, p_i) + X_i, \quad X_i = \begin{cases} x_i, & i \in I \\ 0, & i \notin I \end{cases} \quad (3.1)$$

calculated sequentially for $i = 1, 2, \dots, N$. The outputs of the original neural model y will be the neuron responses at the output neurons, i.e., $y_i = z_k, k = i + N - N_y, k \in K$.

Now we introduce the adjoint neural model, which consists of N adjoint neurons. Let $\hat{z}_j^k$ be the response of the j^{th} adjoint neuron. We interpret $\hat{z}_j^k$ as the gradient of original neural model output w.r.t the local response of the j^{th} neuron in the original neural model, i.e.,

$$\hat{z}_j^k = \frac{\partial z_k}{\partial z_j} \quad (3.2)$$

where $k, k \in K$, indicates an output neuron of interest for which sensitivity is to be computed. In most of the following presentation, we use $\hat{z}_j$ to represent $\hat{z}_j^k$ for simplicity.

The processing function for this adjoint neuron is defined as a linear function,

$$\hat{z}_j = \sum_{i=j+1}^N \frac{\partial f_i}{\partial z_j} \cdot \hat{z}_i + \delta_{kj}, \quad \delta_{kj} = \begin{cases} 1, & k = j \\ 0, & k \neq j \end{cases} \quad (3.3)$$

where $\frac{\partial f_i}{\partial z_j}$, which could be derivatives from microwave empirical functions, are the local derivatives of original neuron functions.

Let J be the Jacobian matrix $(\frac{\partial f^T}{\partial z})^T$, where $f = [f_1, f_2, \dots, f_N]$. For generic feedforward neural networks with neuron indices numbered consecutively starting from the input neurons, through hidden neurons to the outputs neurons, we have $\frac{\partial f_i}{\partial z_j} = 0$ if $j \geq i$.

Equation (3.3) is equivalent to

$$(I - J)^T \cdot \hat{z} = [\delta_{k1} \quad \delta_{k2} \quad \dots \quad \delta_{kN}]^T \quad (3.4)$$

where I is a $N \times N$ identity matrix. Because $(I - J)^T$ is upper diagonal, to perform “feedforward” computation in the adjoint model, we first initialize the last several adjoint

neurons (corresponding to the output neurons in the original neural model) by Kronecker functions, $\hat{z}_j = \delta_{kj}$, $j \in K$. Then we calculate (3.3) *backwards* according to the neuron sequence $j = N-1, N-2, \dots, 1$ without solving equations. The final desired sensitivity solution of the original x - y model can now be obtained explicitly from the adjoint model as $\frac{\partial y_i}{\partial x_j} = \frac{\partial z_k}{\partial z_j} = \hat{z}_j$, $k \in K$, $j \in I$, $i = k + N_y - N$. Notice that the adjoint neurons receiving nonzero external excitation (i.e., δ_{kj}) correspond to the output neurons in the original neural model, i.e., $j = k \in K$.

3.2.2 Basic Adjoint Neural Model Structure

As formulated in (3.3), the input (output) neurons in the adjoint model correspond to the output (input) neurons in the original model, and the sequence of the neuron processing in the adjoint model is exactly the reverse of that in the original neural model. With this concept, a basic adjoint neural network structure can be created by flipping the original neural model between input and output. The connections between the adjoint neurons i and j has a weight value equal to $\partial f_j / \partial z_i$ (to be referred to as local derivative), and processing functions for all adjoint neurons are linear.

Here we use an example to show how to setup a basic adjoint neural model from a given original neural model. The original model is given in Figure 3.2(a). The total number of neurons in the original model is $N = 5$. Knowing that the adjoint neural model is the “reverse” version of original neural model, we realize the adjoint model structure shown in Figure 3.2(b),

Figure 3.2. An example illustrating (a) original neural model and (b) basic adjoint neural model for sensitivity analysis. The input (output) neurons in the adjoint model correspond to the output (input) neurons in the original model. The neuron processing sequence in adjoint model is the reverse of that in the original model.

where, from (3.3) we have,

$$\hat{z}_3 = \hat{z}_4 \cdot \frac{\partial f_4}{\partial z_3} + \hat{z}_5 \cdot \frac{\partial f_5}{\partial z_3} \quad \hat{z}_2 = \hat{z}_3 \cdot \frac{\partial f_3}{\partial z_2} + \hat{z}_4 \cdot \frac{\partial f_4}{\partial z_2} \quad \hat{z}_1 = \hat{z}_2 \cdot \frac{\partial f_2}{\partial z_1} + \hat{z}_3 \cdot \frac{\partial f_3}{\partial z_1} \quad (3.5)$$

By providing values of $\frac{\partial f_4}{\partial z_3}, \frac{\partial f_5}{\partial z_3}, \frac{\partial f_4}{\partial z_2}, \frac{\partial f_3}{\partial z_2}, \frac{\partial f_3}{\partial z_1}, \frac{\partial f_2}{\partial z_1}$ as the connection weights in Figure

3.2(b), we obtain the basic adjoint neural network.

This basic adjoint neural model can be used for first-order sensitivity analysis and for optimization such as physical/geometry optimization of EM problems. When the neural model structure is multilayer perceptron, our technique becomes equivalent to the existing sensitivities in [147][148]. Our method described above expands sensitivity analysis to general microwave neural models such as knowledge based neural models embedding microwave empirical information.

3.2.3 Trainable Adjoint Neural Model Structure

Here we consider a novel and advanced neural modeling requirement, i.e., to use sensitivity as target data for learning. This can be useful for enhancing the reliability of models and for addressing challenges in microwave modeling involving different domains, e.g., large-signal versus small-signal domains because small signal parameters embed the derivative information of large signal model. Here we propose to train adjoint neural models to achieve this task.

If the adjoint neural model is to be trained, the connection weights in the adjoint neural model will vary with respect to (dependent upon) training parameters in both the adjoint and original models. In order to derive a training technique using the neural model framework,

we add a set of fictitious neurons, called Element Derivative Neurons (EDN) whose processing functions are exactly the local derivatives $\partial f_j / \partial z_i$. These EDNs are stimulated by (dependent upon) neurons in the original neural model, and the responses of the EDNs become the stimulus to the adjoint neural model. In general, the EDNs can be created from each neuron in the original neural model shown in Figure 3.3. The EDNs share the same stimuli and parameters as their corresponding original neurons.

The overall sensitivity analysis framework is shown in Figure 3.4 including the original model, the adjoint model and the EDNs, where $\{x_1 x_2 \cdots x_{N_x}\}$, $\{y_1 y_2 \cdots y_{N_y}\}$ and $\{\hat{x}_1 \hat{x}_2 \cdots \hat{x}_{N_x}\}$, $\{\hat{y}_1 \hat{y}_2 \cdots \hat{y}_{N_y}\}$ are the inputs and outputs of the original and adjoint neural models, respectively.

For the adjoint model, the relationships between inputs and outputs are decided in Table 3.1.

Here we use the example from Figure 3.2 to show the setup of a trainable adjoint neural model from the given original neural model. The EDNs are created from the original model shown in Figure 3.5(a) and are connected to the adjoint neural model illustrated in Figure 3.5(b), where the EDNs are defined as,

$$z_{3''} = \frac{\partial f_3}{\partial z_2} \quad z_{3'} = \frac{\partial f_3}{\partial z_1} \quad z_{2'} = \frac{\partial f_2}{\partial z_1} \quad z_{4'} = \frac{\partial f_4}{\partial z_2} \quad z_{4''} = \frac{\partial f_4}{\partial z_3} \quad z_{5'} = \frac{\partial f_5}{\partial z_3}$$

3.2.4 Combined Training of the Adjoint and the Original Neural Models

Let g_j^k represent the derivative training data (i.e., desired target value) for the derivative

$\frac{dz_k}{dx_j}$, $k \in K$. Let g represent the derivative training data including g_j^k for all $k \in K$, $j \in I$.

Figure 3.3. Relationship between the i^{th} original neuron and the fictitious Element Derivative Neurons (EDNs).

Figure 3.4. Original neural model, adjoint neural model and EDNs. The adjoint model in this setup is trainable.

Table 3.1. Relationship between inputs and outputs of adjoint neural model. Inputs to the adjoint model are unit excitations applied to an adjoint input neuron which corresponds to an output neuron in original neural model.

Input $\hat{x}$	Output $\hat{y}$
$[1 \ 0 \ 0 \ \dots \ 0]$	$\begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \dots & \frac{\partial y_1}{\partial x_{N_x}} \end{bmatrix}$
$[0 \ 1 \ 0 \ \dots \ 0]$	$\begin{bmatrix} \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & \dots & \frac{\partial y_2}{\partial x_{N_x}} \end{bmatrix}$
$\begin{matrix} [0 \ \dots \ 1 \ \dots \ 0] \\ 1 \ 2 \ \dots \ k \ \dots \ N_y \end{matrix}$	$\begin{bmatrix} \frac{\partial y_k}{\partial x_1} & \frac{\partial y_k}{\partial x_2} & \dots & \frac{\partial y_k}{\partial x_{N_x}} \end{bmatrix}$
$[0 \ 0 \ 0 \ \dots \ 1]$	$\begin{bmatrix} \frac{\partial y_{N_y}}{\partial x_1} & \frac{\partial y_{N_y}}{\partial x_2} & \dots & \frac{\partial y_{N_y}}{\partial x_{N_x}} \end{bmatrix}$

(a)

Figure 3.5. (a) Illustration of the original neural model and EDNs created, for the example in Figure 3.4. The single or double prime denote EDNs, e.g. $4'$ and $4''$ represent Element Derivative Neurons created from neuron 4.

(b)

Figure 3.5. (b) Illustration of EDNs and trainable adjoint model for the example in Figure 3.4. The single or double prime denote EDNs, e.g., $4'$ and $4''$ represent Element Derivative Neurons created from neuron 4 and used in the adjoint neural model.

We formulate a new training task such that the neural model $y(x, w)$ fits not only the x - y relationship, but simultaneously also the required derivative relationship of y w.r.t x .

To achieve this goal, we utilize the adjoint neural network model such that the training task becomes simultaneous training of the original and the adjoint neural models. Let the per sample training error as functions of w be defined as

$$\tilde{E} = \tilde{E}_o + \tilde{E}_a = \frac{1}{2} \left[W_1 \sum_{k \in K} (z_k - d_k)^2 + W_2 \sum_{i \in I, k \in K} \left(\frac{dz_k}{dx_i} - g_i^k \right)^2 \right] \quad (3.6)$$

where $\tilde{E}_o$ and $\tilde{E}_a$ represent training errors from original and adjoint neural models respectively, d and g represent the training data for the original outputs and their derivatives, subscripts i and k (used for x , z , d and g) indicate original input neuron i and original output neuron k , respectively, and W_1 , W_2 are weights used to balance emphasis between training original and adjoint models. We also call $\tilde{E}_o$ and $\tilde{E}_a$ as original training error and adjoint training error, respectively. The overall training error will be this per sample error $\tilde{E}$ accumulated over various samples in training data set. During training, both the original and the adjoint neural models share a common set of parameters p_i , $i = 1, 2, \dots, N$ to ensure consistency between original and adjoint models, and/or to ensure that training original and adjoint models reinforce each other's accuracy.

Our formulation can accommodate three types of training situations. (i) Train original neural model using input/output data (x, d) . After training, the outputs of adjoint model automatically become explicit derivative of the original input/output relationship. (ii) Train adjoint model to learn derivative data (x, g) . The original model will then give original input/output (i.e., x - y) relationship, which has the effect of providing integration solution

over derivative training data. (iii) Train both original and adjoint models together to learn (x, d) and (x, g) data which will help the neural model to be trained more accurately and robustly. Figure 3.6 shows those types of training.

Figure 3.6. (a) Training to learn original (x, y) input-output relationship, i.e., to learn from data (x, d) . After training, the adjoint model automatically provides explicit sensitivity information of y versus x .

Figure 3.6. (b) Training to learn derivative information of y w.r.t x , i.e., to learn from data (x, g) . After training, the original model provides (x, y) relationship with integration effect on training data g .

Figure 3.6. (c) Training to simultaneously learn both input-output and its derivative information, enhancing reliability of the neural model.

We can achieve these three training cases using our general formulation of (3.6) by setting

(i) $W_2 = 0$, or (ii) $W_1 = 0$ or (iii) $W_1 \neq 0$ and $W_2 \neq 0$, respectively.

3.2.5 Second-order Sensitivity Analysis

Training is to adjust neural network internal parameters p_i for each neuron such that the accumulated training error of $\tilde{E}$ is minimized. The training algorithms, such as conjugate gradient method, quasi-Newton method and Backpropagation [1] typically require the derivative of $\tilde{E}$ w.r.t p_i .

First considering the training errors due to training of the adjoint neural model to learn input/output derivative data, it becomes necessary to perform second-order sensitivity analysis.

Let

$$\tilde{E}_a = \sum_{k \in K} \tilde{E}_{a,k} \quad (3.7)$$

be defined as derivative training error for each sample data, where $\tilde{E}_{a,k}$ is the training error between adjoint model and the sensitivity data for the k^{th} output neuron in the original model. Let ψ_i represent an element in vector p_i , which are the parameters of the i^{th} neuron in the original model. To find the derivatives required to train the adjoint model for each sample, we first differentiate $\tilde{E}_{a,k}$ as

$$\frac{\partial \tilde{E}_{a,k}}{\partial \psi_i} = \sum_{l \in I} W_2 \cdot (\hat{z}_l - g_{kl}) \cdot \frac{\partial \hat{z}_l}{\partial \psi_i} = G^T \cdot \frac{\partial \hat{z}}{\partial \psi_i} \quad (3.8)$$

where G is a column vector of size N with elements,

$$G_i = \begin{cases} W_2 \cdot (\hat{z}_i - g_{ki}), & i \in I \\ 0, & i \notin I \end{cases}$$

which is the training error at the output neuron in the adjoint model, i.e., adjoint neuron i . To

obtain $\frac{\partial \hat{z}}{\partial \psi_i}$, we can differentiate (3.4) with respect to parameter ψ_i as

$$(I - J)^T \cdot \frac{\partial \hat{z}}{\partial \psi_i} - \left(\frac{\partial J}{\partial \psi_i} + \sum_{n=i+1}^N \frac{\partial J}{\partial z_n} \cdot \frac{\partial z_n}{\partial \psi_i} \right)^T \cdot \hat{z} = 0 \quad (3.9)$$

Now (3.8) can be replaced by

$$\frac{\partial \tilde{E}_{a,k}}{\partial \psi_i} = G^T \cdot [(I - J)^{-1}]^T \cdot \left(\frac{\partial J}{\partial \psi_i} + \sum_{n=i+1}^N \frac{\partial J}{\partial z_n} \cdot \frac{\partial f_n}{\partial \psi_i} \right)^T \cdot \hat{z} \quad (3.10)$$

Let $\hat{\hat{z}}$ be defined as a vector solution for

$$(I - J) \cdot \hat{\hat{z}} = G \quad (3.11)$$

$\hat{\hat{z}}$ can be interpreted as error propagation signal in adjoint neural model, which is solved from back-propagation in the adjoint model according to the neuron processing sequence $j = 2, 3, \dots, N$ by initializing $\hat{\hat{z}}_1 = W_2 \cdot (\hat{z}_1 - g_{k1})$, and

$$\hat{\hat{z}}_j = \sum_{m=1}^{j-1} \frac{\partial f_j}{\partial z_m} \cdot \hat{\hat{z}}_m + G_j \quad (3.12)$$

Equation (3.10) now becomes,

$$\begin{aligned} \frac{\partial \tilde{E}_{a,k}}{\partial \psi_i} &= \hat{\hat{z}}^T \cdot \left(\frac{\partial J}{\partial \psi_i} + \sum_{n=i+1}^N \frac{\partial J}{\partial z_n} \cdot \frac{\partial z_n}{\partial \psi_i} \right)^T \cdot \hat{z} \\ &= \sum_{j=1}^{i-1} \hat{\hat{z}}_j \cdot \frac{\partial^2 f_i}{\partial z_j \partial \psi_i} \cdot \hat{z}_i + \sum_{n=i+1}^N \left(\sum_{j=1}^{N-1} \sum_{m=j+1}^N \hat{\hat{z}}_j \cdot \frac{\partial^2 f_m}{\partial z_j \partial z_n} \cdot \hat{z}_m \right) \cdot \frac{\partial z_n}{\partial \psi_i} \end{aligned} \quad (3.13)$$

where $\frac{\partial^2 f_i}{\partial z_j \partial \psi_i}$ represents second-order derivative information in the individual neurons.

Next, we define a new backpropagation from the adjoint neural model into the original neural model through EDNs as

$$\bar{\hat{z}}_n = \sum_{j=1}^{N-1} \sum_{\substack{m= \\ \max(j+1, n+1)}}^N \hat{\hat{z}}_j \frac{\partial^2 f_m}{\partial z_j \partial z_n} \cdot \hat{z}_m \quad (3.14)$$

The last term in (3.13) can be handled by injecting $\bar{\hat{z}}_n$ into the original neural model as an additional error propagation to be merged together with error propagation in the original model. Notice that $\hat{z}$, $\hat{\hat{z}}$ and $\bar{\hat{z}}$ are all defined corresponding to the sensitivity of selected neuron k as in (3.2), which means $\hat{z}_j = \hat{z}_j^k$, $\hat{\hat{z}}_j = \hat{\hat{z}}_j^k$, $\bar{\hat{z}}_j = \bar{\hat{z}}_j^k$.

Now we include $\tilde{E}_o$ to consider the derivative for the total training error per sample of (3.6).

Utilizing (3.13) and (3.14), we have,

$$\begin{aligned} \frac{\partial \tilde{E}}{\partial p_i} &= \frac{\partial \tilde{E}_o}{\partial p_i} + \frac{\partial \tilde{E}_a}{\partial p_i} \\ &= \sum_{k \in K} [W_1 \cdot (z_k - d_k) \cdot \frac{\partial z_k}{\partial z_i} \cdot \frac{\partial f_i}{\partial p_i} + \sum_{n=i+1}^N \bar{\hat{z}}_n \cdot \frac{\partial z_n}{\partial z_i} \cdot \frac{\partial f_i}{\partial p_i} + \sum_{j=1}^{i-1} \hat{\hat{z}}_j \cdot \frac{\partial^2 f_i}{\partial z_j \partial p_i} \cdot \hat{z}_i^k] \end{aligned} \quad (3.15)$$

where $\tilde{E}_o$ is the original training error for each sample data.

According to (3.15), there are three concurrent backpropagation paths in our task, corresponding to the three terms in the equation. The first path is that of the training error in the original network, i.e., $W_1 \cdot (z_j - d_j)$ $j \in K$, which starts from the output neurons in the original model, backpropagates through the original hidden neurons towards the original

input neurons. The second path is that of the adjoint training error, i.e., $G_i = W_2 \cdot (\hat{z}_i - g_{ki})$, $i \in I$, which starts from the adjoint output neurons through the EDNs, and into the original neural network towards the original input neurons. The third path is that of the adjoint training error, i.e., G_i , $i \in I$, which starts from output neurons in the adjoint neural model and backpropagates towards the EDNs.

To formulate our training into an efficient and concurrent original/adjoint neural network backpropagation scheme, we further process the first and second paths as follows.

Let $\bar{\sigma}_j = \frac{\partial \tilde{E}}{\partial z_j}$ be the new combined local gradient representing original and derivative training error backpropagated to neural j in the original model, i.e., the backpropagation of path 1 and path 2 merged together at neuron j in the original neural model. This combined backpropagation continues towards original input neurons, merges again with $\bar{z}$ (which is the backpropagation from adjoint model through EDNs to original model) arriving at every neuron the combined backpropagation encounters along the way:

$$\bar{\sigma}_j = \sum_{m=j+1}^N \bar{\sigma}_m \cdot \frac{\partial f_m}{\partial z_j} + D_j + \sum_{k \in K} \bar{z}_j^k, \quad D_j = \begin{cases} W_1 \cdot (z_j - d_j), & j \in K \\ 0, & j \notin K \end{cases} \quad (3.16)$$

where D_j is the training error for backpropagation path 1 at the original output neurons.

Then, the derivative required by training due to the first two parts in (3.15) is: $\bar{\sigma}_i \cdot \frac{\partial f_i}{\partial p_i}$.

Now the final derivative for training the combined original and adjoint model is,

$$\frac{\partial \tilde{E}}{\partial p_i} = \bar{\sigma}_i \cdot \frac{\partial f_i}{\partial p_i} + \sum_{k \in K} \sum_{j=1}^{i-1} \hat{z}_j^k \cdot \frac{\partial^2 f_i}{\partial z_j \partial p_i} \cdot \hat{z}_i^k \quad (3.17)$$

which includes first- and second-order derivatives. Notice that even though the derivation process is complicated, the final result of (3.17) is surprisingly simple and elegant, fully compatible with neural network concept of error propagation. Also notice that first-order sensitivity analysis in subsection 3.2.1 requires only one backpropagation, whereas the combined first- and second-order sensitivity technique of (3.17) requires three error propagation paths with paths 1 and 2 merged as the propagation continues along the way. The proposed method is suitable for incorporation into microwave neural modeling software.

3.3 Demonstration Examples

3.3.1 Example A: High-speed VLSI Interconnect Modeling and Optimization

Fast and accurate sensitivity analysis of coupled transmission lines is important for high-speed VLSI interconnect optimization [89] and statistical design. This example illustrates the proposed sensitivity technique for an arbitrary neural network structure where microstrip empirical formulas are used as part of a knowledge based neural network structure shown in Figure 3.7(a). The inputs to our model (x) are conductor widths (w_1, w_2), spacing between coupled interconnects (s), substrate thickness (h), dielectric constant (ϵ_r), and frequency (f). The output of model (y) is mutual inductance L_{12} .

After training the original model of Figure 3.7(a) using *NeuroModeler* [8] with accurate EM based microstrip data (100 samples) obtained by *LINPAR* [149], we use the proposed method to provide exact derivatives of electrical parameters of the transmission line with respect to

the physical-geometrical parameters needed in VLSI interconnect optimization. The sensitivity solution from the basic adjoint neural model of Figure 3.7(b) is verified by central difference perturbation method in Table 3.2. Figure 3.8 compares our sensitivity versus that from perturbation as a continuous function in s and h sub-spaces respectively. The good agreement in those figures verifies our adjoint model. Notice that the exact sensitivity is obtained through the adjoint neural model without extra training. Without the neural model, such sensitivity would have been computed in EM simulators by perturbation. The computation time for the proposed method compared to EM perturbation solution is 3s versus 2660s for sensitivity analysis of 1000 microstrip models, which are typically needed in the optimization of a network of VLSI interconnects.

Now we consider an advanced use of the neural model just trained. The purpose is to find the solution of feasible regions of interconnect geometry (x of neural model) from given budget on electrical parameters (y of neural model). This is also called design solution space analysis, which is very useful for synthesis of VLSI interconnects and for making trade-off decisions during early design stages of VLSI systems. A basic step is to use optimization to find inputs x of the neuron model from given specifications on y . The overall solution space is solved by repeatedly perform such optimization for a variety of y specifications and a variety of x patterns.

Figure 3.9 shows a solution of feasible space of s versus h for various given design budget of mutual inductance L_{12} . This solution is obtained with 40 optimizations of the trained neural model and the gradient information required by optimization are provided by the adjoint model of Figure 3.7(b).

Figure 3.7. (a) Knowledge based coupled transmission line neural model of mutual inductance (L_{12}) for VLSI interconnect optimization. w_1 , w_2 , s , h , ϵ_r , and f are conductor widths, spacing between coupled interconnects, substrate thickness, dielectric constant and frequency, respectively.

$$\frac{dL_{12}}{dw_1} \quad \frac{dL_{12}}{dw_2} \quad \frac{dL_{12}}{ds} \quad \frac{dL_{12}}{dh} \quad \frac{dL_{12}}{d\epsilon_r} \quad \frac{dL_{12}}{df}$$

(b)

Figure 3.7. (b) Basic adjoint neural model, which will be used by optimization to perform solution space analysis and synthesis of this coupled transmission line.

Table 3.2: Example of sensitivity between perturbation technique and adjoint technique for the VLSI interconnet modeling example.

Good agreement is achieved.

Sensitivity	Perturbation Technique	Adjoint Technique	Difference (%)
dL_{12}/dw_1	-0.1440	-0.1435	0.354
dL_{12}/dw_2	0.0620	0.0616	0.645
dL_{12}/ds	-0.8462	-0.8514	0.610
dL_{12}/dh	0.5338	0.5337	0.018
$dL_{12}/d\epsilon_r$	-0.0010	-0.0010	0.001
$dL_{12}/dfreq$	-0.0037	-0.0037	0.001

Figure 3.8. Sensitivity verification for the VLSI interconnect modeling example (a) dL_{12}/dw_1 versus s (b) dL_{12}/ds versus h . Good agreement is observed between our sensitivity solution and EM perturbation sensitivity.

Figure 3.9. Solution space analysis: feasible regions of s - h of VLSI interconnect design for given design budgets on L_{12} . This solution space is obtained after 40 separate optimizations, where gradient information required by optimization is supplied by the adjoint model of Figure 3.7(b).

3.3.2 Example B: Nonlinear Charge Modeling

This example illustrates the integration effect of the adjoint neural model. We first train only the adjoint neural model to learn the nonlinear capacitor data, which is generated from *Agilent-ADS* [128]. After training with 41 data samples, we perform the testing by comparing the output of the adjoint neural model with different set of nonlinear capacitor data never used in training, as shown in Figure 3.10, where excellent agreement is achieved.

Figure 3.10. Comparison of C between the adjoint neural model and nonlinear capacitor data generated from *Agilent-ADS*. Good agreement is achieved even though such data were never used in training.

We then use the original neural model without re-training (with internal parameters updated according Section 3.2.5) as a nonlinear charge-model (i.e., Q-model). The charge model is compared with analytical integration of ADS capacitor formula (Figure 3.11).

Figure 3.11. Comparison of charge model trained from nonlinear capacitance data with that from analytical integration of *ADS* capacitance formula. Training was done by training the adjoint neural model from capacitance data. After training, the original neural model automatically produces the charge model achieving integration effect of training data. The charge model for nonlinear capacitors is useful for harmonic balance simulation.

The good agreement in the figure verifies the integration effect of training the adjoint neural model. This example shows an interesting solution to one of the frequently encountered obstacles in developing a charge model for nonlinear capacitors required for harmonic balance simulators with only capacitor data available.

3.3.3 Example C: Large-signal FET Modeling

This example shows large-signal device modeling using DC and small-signal training data. The model used is a knowledge based approach where existing intrinsic electrical equivalent circuit model is combined with neural network learning. In practice, manually creating formulas for the nonlinear currents and charge sources in a FET model could be very time-consuming. Here we use neural networks to automatically learn the unknown relationship of gate-source charge Q_{gs} , gate-drain current I_{gd} and drain-source current I_{ds} as nonlinear functions of gate-source and drain-source voltages, V_{gs} and V_{ds} , respectively. However we do not have explicitly the charge data Q_{gs} and dynamic currents data I_{gd} and I_{ds} for training the model. The available training data is the DC and bias-dependent S -parameters of the overall FET, which in our example is generated using *Agilent-ADS* with the Statz Model [113]. Therefore the neural models and the rest of the FET equivalent circuit are combined into a knowledge based model and they together are trained to learn the training data, shown in Figure 3.12. Both S -parameter data and all the DC bias data are used for simultaneous training involving all the original and adjoint neural models. Notice that learning S -parameters means learning the derivative information of the large-signal model. After training, a good agreement of DC and small signal responses at all the 90 bias points between our knowledge based neural FET model and those given by the ADS solution is observed, as shown in Figures 3.13-3.14.

Figure 3.12. Large-signal FET modeling including adjoint neural networks trained by DC and bias-dependent S -parameters. Here the adjoint neural networks complement an intrinsic FET equivalent circuit by providing the unknown nonlinear currents (I_{ds} , I_{gd}) and charge (Q_{gs}). The small-signal S -parameters imply the derivative information of the large signal model. This example shows combined training of original neural model to learn DC data and simultaneously adjoint neural model to learn small-signal S -parameter data. Microwave knowledge of a basic equivalent circuit is combined with sub-neural models leading to knowledge based approach for FET modeling.

Figure 3.13. Comparison between DC curves of the ADS Statz model (—) and our knowledge based neural FET model (o).

Figure 3.14. Comparison between S -parameters of the ADS Statz model (—) and our knowledge based neural FET model at four of the ninety bias points : (a) $\{V_{ds} = 3.26 \text{ V}, V_{gs} = -0.6 \text{ V}\}$ and $\{V_{ds} = 0.26 \text{ V}, V_{gs} = -0.6 \text{ V}\}$, (b) $\{V_{ds} = 0.9 \text{ V}, V_{gs} = -0.6 \text{ V}\}$ and $\{V_{ds} = 0.9 \text{ V}, V_{gs} = 0.0 \text{ V}\}$.

We then used our complete knowledge based neural FET model, as shown in Figure 3.15, in a three-stage power amplifier shown in Figure 3.16. for large-signal harmonic balance simulation. The large-signal response of the amplifier using our model agrees well with that using original ADS model illustrated in Figure 3.17.

Figure 3.15. Complete knowledge based neural FET model, where $R_g = 4.0 \, \Omega$, $R_s = 4.8994 \, \Omega$, $R_d = 0.05 \, \Omega$, $L_g = 0.3167 \, \text{nH}$, $L_s = 0.088 \, \text{nH}$, $L_d = 0.1966 \, \text{nH}$, $R_x = 794.235 \, \Omega$, $C_x = 20.0 \, \text{pF}$, and $C_{ds} = 0.09916 \, \text{pF}$ are extrinsic components.

Figure 3.16. The 3-stage amplifier where the FET models used are knowledge based neural FET models trained from the proposed method following Figure 3.12.

Figure 3.17. Comparison of the power amplifier large-signal responses (a) Time domain amplifier responses using the ADS Statz model and our knowledge based neural FET model, (b) Output spectrum of the amplifier using the ADS Statz model and our model. The neural model trained with DC and S -parameter data is used here for harmonic balance based amplifier design, made possible by our proposed approach of training the adjoint model.

Our example demonstrates the capability of the adjoint neural networks in enhancing conventional FET models through adding *trainable* nonlinear current or charge relationships to the model. Such trainable nonlinear relationship is especially beneficial when analytical formulas in the FET problem is unknown or available formulas are not suitable. By combining adjoint neural networks with the existing FET models, one can improve the models efficiently without having to go through the trial and error process typically needed during manual creation of empirical functions. The proposed method provides a new alternative for efficient generation of nonlinear device models for use in large-signal simulation and design.

3.4 Summary

This chapter presented a unified framework for neural based modeling and sensitivity analysis for generic types of microwave neural models including knowledge based models. The proposed method provides continuous and differentiable models with analytically consistent derivatives from raw information present in the original training data. A novel and elegant first- and second-order sensitivity analysis scheme allows the training of neural models to learn not only input-output relationships in a microwave component but also its derivatives. This leads to a major and important application of the ADJNN technique, i.e., efficient and accurate nonlinear microwave device and circuit modeling. The ADJNN approach uses a combination of circuit and neural models, where the circuit dynamics are defined by the topology and the nonlinearity is defined by ANNs. The circuit topology can be obtained from empirical models or equivalent circuits. Using the ADJNN technique, such neural based nonlinear device/circuit model can be developed using DC and small-signal

data. The trained model can be subsequently used to predict large-signal effects in microwave circuit or system design.

CHAPTER 4

Dynamic Neural Network Technique for Microwave Modeling

In this chapter, a major contribution of this thesis namely, the Dynamic Neural Network (DNN) modeling technique [5][6] is presented. Compared to the ADJNN modeling technique described in previous chapter, the DNN technique models the complete dynamic and nonlinear behavioral of the nonlinear microwave device/circuit in the absence of existing knowledge of such device or circuit.

The proposed DNN model is achieved in the most desirable format, i.e., continuous time-domain dynamic system format. The DNN can be developed directly from input-output data without having to rely on internal details of the device or circuit. An algorithm is developed to train the model with time or frequency domain large signal information. Efficient representations of the model are proposed for convenient incorporation of DNN into high-level circuit or system simulation. The proposed DNN retains or enhances the advantages of learning, speed, and accuracy as in existing neural network techniques; and provides additional advantages of being theoretically elegant and practically suitable for diverse needs of nonlinear microwave simulation, e.g., standardized implementation in simulators, suitability for both time and frequency domain applications, and multi-tone simulations.

4.1 Introduction

This chapter addresses an important application of ANN, i.e., application to nonlinear circuit modeling and design. This could be a significant area because of the increasing need for efficient CAD algorithms in high-level and large-scale nonlinear microwave design. Recently, several ANN methods were introduced with emphasis on nonlinear circuit modeling, such as the neural network-based behavioral model [127][139] and discrete recurrent neural network [55][150] approaches. These works demonstrated neural networks as a useful alternative to the conventional behavioral or equivalent circuit based approaches [125][126][129][131][133][135][136]. The neural network method in [139] is formulated to overcome the limitations in conventional behavioral models by providing bidirectional behavior allowing more accurate system simulation. The recurrent neural network approach [55] achieves a discrete time domain model based on backpropagation-through-time training to learn the circuit input-output relationship. However, because of the specific formats of these existing neural based methods, there still exist limitations due to difficulties in their incorporation in existing nonlinear simulators, in establishing relations with large-signal measurement, limited flexibility for different simulations, or the curse of dimensionality in multi-tone simulations.

The most ideal format to describe nonlinear dynamic models for the purpose of circuit simulation is the continuous time-domain format, e.g., the popularly accepted dynamic current-charge format in many harmonic balance simulators. This format in theory best describes the fundamental essence of nonlinear behavior, and in practice is most flexible to fit most or nearly all needs of nonlinear microwave simulation, a task not yet achieved by the

existing ANN-based techniques. In the neural network community, such type of networks has been studied, e.g., Hopfield network [151], recurrent network [2], etc. However they were mainly oriented for digital signal processing such as binary-based image processing [2], or system control with online correction signals from a physical system [152]. They are not directly suitable for microwave modeling. We must address continuous analog signals and our CAD method must be able to predict circuit behavior off-line.

For the first time, an exactly continuous time-domain dynamic-modeling method is formulated using neural networks for large-signal modeling of nonlinear microwave circuits and systems [5][6]. The model, called dynamic neural network (DNN) model, can be developed directly from input-output data without having to rely on internal details of the circuits. An algorithm is described to train the model with time or frequency domain information. Efficient representations of DNN are proposed such that the model can be conveniently incorporated into circuit simulators for high-level and large-scale nonlinear microwave design. The model can be standardized even with diverse requirements of nonlinear modeling such as single- and multi-tone applications, and training with time- or frequency-domain data.

4.2 Dynamic Neural Network Modeling of Nonlinear Circuits: Formulation and Development

4.2.1 Original Circuit Dynamics

Let $\mathbf{u} = [u_1 \ u_2 \ \dots \ u_{N_u}]^T$ be vector of the input signals of the nonlinear circuit, where N_u is the

number of inputs. Within this chapter, we use $y = [y_1 \ y_2 \ \dots \ y_{N_y}]^T$ to represent the vector of the output signals of the nonlinear circuit, where N_y is the number of outputs. The original nonlinear circuit can be generally described in state equation form as

$$\begin{aligned}\dot{\bar{v}}(t) &= \varphi(\bar{v}(t), u(t)) \\ y(t) &= \psi(\bar{v}(t), u(t))\end{aligned}\tag{4.1}$$

where $\bar{v}$ is a N_S -vector of state variables and N_S is the number of states, φ and ψ represent nonlinear functions. In a modified nodal formulation [153], the state vector $\bar{v}(t)$ includes nodal voltages, currents of inductors, currents of voltage sources and charge of nonlinear capacitors.

For a circuit with many components, (4.1) could be a large set of nonlinear differential equations. For system level simulation including many circuits, such detailed state equations are too large, computationally expensive, and sometimes even unavailable at system level. Therefore, a simpler (reduced order) model approximating the same dynamic input-output relationships is needed.

4.2.2 Formulation of Dynamic Neural Network (DNN) Model

Let n be the order of the reduced model, $n < N_S$. Let $y^{(i)}(t) = d^i y(t)/dt^i$ and $u^{(i)}(t) = d^i u(t)/dt^i$ denote the i^{th} order derivatives of $y(t)$ and $u(t)$ with respect to t , respectively. In order to derive a dynamic model, the original problem (4.1) is reformulated into reduced order differential equations using the input-output variables as

$$y^{(n)}(t) = f(y^{(n-1)}(t), y^{(n-2)}(t), \dots, y(t), u^{(n)}(t), u^{(n-1)}(t), \dots, u(t))\tag{4.2}$$

where f represents nonlinear functions. Here, we propose to employ the ANN to represent the nonlinear relationships between the dynamic information of inputs and outputs. The

schematic of the proposed DNN model is shown in Figure 4.1.

Figure 4.1. Schematic of Dynamic Neural Network (DNN) approach for nonlinear circuit modeling in continuous time domain.

Let $\hat{v}_i$ be a N_y -vector, $i = 1, 2, \dots, n$. Let f_{ANN} represent a multilayer perceptron neural network [1] with input neurons representing y, u , their derivatives $d^i y/dt^i, i=1, 2, \dots, n-1$, and $d^k u/dt^k, k=1, 2, \dots, n$; and the output neuron representing $d^n y/dt^n$. The proposed DNN model is derived from (4.2) as

$$\begin{aligned} \dot{\hat{v}}_1(t) &= \hat{v}_2(t) \\ &\vdots \\ \dot{\hat{v}}_{n-1}(t) &= \hat{v}_n(t) \\ \dot{\hat{v}}_n(t) &= f_{ANN}(\hat{v}_n(t), \hat{v}_{n-1}(t), \dots, \hat{v}_1(t), u^{(n)}(t), u^{(n-1)}(t), \dots, u(t)) \end{aligned} \quad (4.3)$$

and the inputs and outputs of the model is $u(t)$ and $y(t) = \hat{v}_1(t)$, respectively.

The overall DNN model (4.3) is in a standardized format for typical nonlinear circuit simulators. For example, the left-hand-side of the equation provides the charge (Q) or the capacitor part, and the right-hand-side provides the current (I) part, which are the standard representation of nonlinear components in many harmonic balance simulators.

The proposed DNN overcomes the limitations of the previous static I-Q neural model of [64] which was only suitable for intrinsic FETs. The proposed DNN can provide dynamic current-charge parameters for general nonlinear circuits with any number of internal nodes in original circuit. The order n (or the number of hidden neurons in f_{ANN}) represents the effective order (or the degree of nonlinearity) of the original circuit that is visible from the input-output data. Therefore the size of the DNN reflects the internal property of the original circuit rather than external signals, and as such the model does not suffer from curse of dimensionality in multi-tone simulation.

The proposed DNN is a generic dynamic model, which can be used in periodic [5][6] or transient [154] simulation. Within this thesis, we consider the training and application of DNN in periodic state case, where harmonic balance simulation is performed.

4.2.3 Model Training

Our DNN model will represent a nonlinear microwave circuit only after we train it with data from the original circuit. We use training data in the form of input/output harmonic spectrums, which can be obtained through simulation or measurements. Let $\bar{U}(\omega)$ and $\bar{Y}(\omega)$ be such input and output spectrums respectively, $\omega \in \Omega$, where Ω is the set of spectrum frequencies. The training data is generated using a variety of input samples, leading

to a set of data $\bar{U}_m(\omega)$ and $\bar{Y}_m(\omega)$, where m is the sample index, $m=1, 2, \dots, N_t$, and N_t is the total number of samples.

A second set of data, called testing data should also be obtained similarly from the original circuit for model verification. The testing data should be generated using a set of input samples different from those used in training data.

Initial Training: We first train the f_{ANN} part of the DNN model in the time domain directly or indirectly using time-domain information. Suppose matrix $A(\omega, t)$ represents the coefficients of Inverse Fourier Transform [155]. Let the derivative of $A(\omega, t)$ w.r.t time t be represented as

$$A^{(i)}(\omega, t) = \frac{\partial A^i(\omega, t)}{\partial t^i} \quad (4.4)$$

The training data for f_{ANN} can be derived from

$$\bar{y}_m^{(i)}(t) = \sum_{\omega \in \Omega} A^{(i)}(\omega, t) \cdot \bar{Y}_m(\omega) \quad (4.5)$$

$$\bar{u}_m^{(i)}(t) = \sum_{\omega \in \Omega} A^{(i)}(\omega, t) \cdot \bar{U}_m(\omega) \quad (4.6)$$

The initial training is illustrated in Figure 4.2. The objective of the training is to adjust ANN internal weight parameters to minimize the error function

$$E_1 = \frac{1}{2} \sum_{t \in T} \sum_{m=1}^{N_t} \left\| f_{ANN}(\bar{y}_m^{(n-1)}(t), \dots, \bar{y}_m(t), \bar{u}_m^{(n)}(t), \dots, \bar{u}_m(t)) - \bar{y}_m^{(n)}(t) \right\|^2 \quad (4.7)$$

where T is the set of time points used by Fourier Transform [155].

This process is computationally efficient (without involving harmonic balance simulation)

and can train the f_{ANN} from a random (unknown) start to an approximate solution. Because all input-output information in each sample of training data are at the same instance of time, this proposed technique is completely free from restrictions on sampling frequencies, representing an clear advantage over the previous discrete recurrent neural network method [55].

Figure 4.2. *Initial training of DNN: to train the f_{ANN} part in the time domain using spectrum data, where $A^{(i)}$ is the time derivative operator corresponding to (4.4).*

Final Training: The DNN model is further refined using results from initial training as starting point. Final training is done in frequency-domain involving HB solutions of the DNN model.

The error function as functions of ANN internal weight parameters for training is,

$$E_2 = \frac{1}{2} \sum_{m=1}^{N_t} \sum_{\omega \in \Omega} \|Y_m(\omega) - \bar{Y}_m(\omega)\|^2 \quad (4.8)$$

where $Y_m(\omega)$ and $\bar{Y}_m(\omega)$ represent spectrum from model and m^{th} sample of training data, respectively. In order to achieve the harmonic solutions $Y_m(\omega)$ from the DNN model, we apply differentiation over the f_{ANN} using the adjoint neural network method described in Chapter 3. The resulting derivatives, i.e., $\frac{\partial f_{ANN}}{\partial y^{(n-1)}}, \dots, \frac{\partial f_{ANN}}{\partial y} \frac{\partial f_{ANN}}{\partial u^{(n)}}, \dots, \frac{\partial f_{ANN}}{\partial u}$, fit the Jacobian matrix of harmonic balance equations, as shown in Figure 4.3.

The training technique presented here demonstrates that both time and frequency domain data can be used for DNN training. The compatibility of DNN training with large-signal harmonic data is an important advantage over the discrete recurrent neural network approach [55] whose training is limited to the time domain only.

4.2.4 Use of The Trained DNN Model in Circuit Simulation

(1) Method 1: Circuit Representation of DNN

An exact circuit representation of our DNN model can be derived as shown in Figure 4.4(a). The state variables are represented by voltages on unit capacitors with their currents controlled by other state variables, e.g., $\hat{C} \cdot \dot{\hat{v}}_1(t) = \hat{v}_2(t)$, where $\hat{C} = 1$. The dynamic model inputs are defined as voltages on unit inductors with their currents controlled by input dynamics of different orders, e.g., $u^{(1)}(t) = \hat{L} \cdot \dot{u}(t)$, where $\hat{L} = 1$. In this way, the trained

Figure 4.3. Evaluation of f_{ANN} and its derivatives required during HB simulation is provided by original and adjoint neural networks, respectively.

model can be conveniently incorporated into available simulation tools for high-level circuit and system design. This can be achieved in most existing simulators without doing computer programming.

(2) *Efficient Harmonic Balance (HB) Representation of DNN*

Here we propose another method for incorporating the DNN model into circuit simulation. We use HB as the circuit simulation environment. Through the formulation described below, we are able to eliminate most of the state variables in DNN by Fourier Transform and use even fewer variables during HB simulation, further speeding-up circuit simulation. The HB representation is shown in Figure 4.4(b).

Let $U(\omega)$ and $Y(\omega)$ be the Fourier Transform of input $u(t)$ and output $y(t)$, respectively.

Let $B(\omega, t)$ represent the Fourier Transform matrix, such that

$$Y(\omega) = \sum_{t \in T} B(\omega, t) \cdot y(t) \quad (4.9)$$

$$U(\omega) = \sum_{t \in T} B(\omega, t) \cdot u(t) \quad (4.10)$$

Since

$$y^{(i)}(t) = \sum_{\omega \in \Omega} A^{(i)}(\omega, t) \cdot Y(\omega) \quad (4.11)$$

$$u^{(i)}(t) = \sum_{\omega \in \Omega} A^{(i)}(\omega, t) \cdot U(\omega) \quad (4.12)$$

pre-multiplying $B(\omega, t)$ to the f_{ANN} equation in DNN model of (4.3), we have the HB equation for the DNN as,

Figure 4.4. Representations of DNN for incorporation into high-level simulation. (a) Circuit representation of the DNN model. (b) HB representation of the DNN model. The two representations are different only in implementation and they are numerically equivalent to each other.

$$\begin{aligned}
& \sum_{t \in T} B(\omega, t) \cdot \sum_{\omega \in \Omega} A^{(n)}(\omega, t) \cdot Y(\omega) - \sum_{t \in T} B(\omega, t) \cdot f_{ANN} \left(\sum_{\omega \in \Omega} A^{(n-1)}(\omega, t) \cdot Y(\omega), \right. \\
& \quad \left. \sum_{\omega \in \Omega} A^{(n-2)}(\omega, t) \cdot Y(\omega), \dots, \sum_{\omega \in \Omega} A(\omega, t) \cdot Y(\omega), \sum_{\omega \in \Omega} A^{(n)}(\omega, t) \cdot U(\omega), \right. \\
& \quad \left. \sum_{\omega \in \Omega} A^{(n-1)}(\omega, t) \cdot U(\omega), \dots, \sum_{\omega \in \Omega} A(\omega, t) \cdot U(\omega) \right) = 0
\end{aligned} \quad (4.13)$$

where $Y(\omega)$ is the Fourier Transform of the time domain signal $y(t)$ as defined earlier.

Substituting Equation (4.11) and (4.12) into the f_{ANN} equation of DNN in (4.3), we have an input-output waveform equation

$$\begin{aligned}
& \sum_{\omega \in \Omega} A^{(n)}(\omega, t) \cdot \sum_{\tau \in T} B(\omega, \tau) \cdot y(\tau) - f_{ANN} \left(\sum_{\omega \in \Omega} A^{(n-1)}(\omega, t) \cdot \sum_{\tau \in T} B(\omega, \tau) \cdot y(\tau), \right. \\
& \quad \left. \sum_{\omega \in \Omega} A^{(n-2)}(\omega, t) \cdot \sum_{\tau \in T} B(\omega, \tau) \cdot y(\tau), \dots, \sum_{\omega \in \Omega} A(\omega, t) \cdot \sum_{\tau \in T} B(\omega, \tau) \cdot y(\tau), \right. \\
& \quad \left. \sum_{\omega \in \Omega} A^{(n)}(\omega, t) \cdot \sum_{\tau \in T} B(\omega, \tau) \cdot u(\tau), \dots, \sum_{\omega \in \Omega} A(\omega, t) \cdot \sum_{\tau \in T} B(\omega, \tau) \cdot u(\tau) \right) = 0
\end{aligned} \quad (4.14)$$

Let $\hat{y}$, $\hat{u}$ be vectors containing $y(t)$ and $u(t)$ for all the time samples t , $t \in T$. Let $\hat{Y}$ and $\hat{U}$ be vectors containing $Y(\omega)$ and $U(\omega)$ at all the spectrum components ω , $\omega \in \Omega$. Since $A(\omega, t)$, $B(\omega, t)$ and $A^{(i)}(\omega, t)$ contain Fourier base functions and their time-derivatives, they are independent of any signals in the circuit and are constants during HB simulation. Therefore, the HB equations for DNN in (4.13) can be expressed as.

$$F(\hat{Y}, \hat{U}) = 0 \quad (4.15)$$

where $F()$ means “nonlinear functions of”. Equation (4.14) can be expressed as

$$H(\hat{y}, \hat{u}) = 0 \quad (4.16)$$

where $H()$ also means “nonlinear functions of”.

We call (4.15) or (4.16) as the HB representation of DNN. To implement DNN into HB circuit simulation, we program either (4.15) or (4.16) within the HB environment. In (4.15), given input harmonic values $\hat{U}$, the DNN will produce output harmonic $\hat{Y}$. In (4.16), given input waveforms $\hat{u}$, the DNN will produce output waveforms $\hat{y}$. Notice that (4.16) uses only $\hat{y}$ and $\hat{u}$ (without explicit derivative variables) at all time points. The HB simulator will solve the overall HB equation including DNN during HB simulation.

In this way, the variables for HB simulation due to DNN are only $\hat{Y}$, $\hat{U}$. All higher-order information of inputs and outputs will be implied by $\hat{Y}$ and $\hat{U}$ through Fourier transformations. Since the total number of nonlinear nodes from the DNN is n times less than that in the circuit representation of DNN, this HB simulation will have further computation speed up.

Notice that (4.15) or (4.16) is only used as an interface when DNN is implemented to circuit simulator. The DNN model itself is the dynamic equation (4.3). Since DNN is a continuous time domain model, the model is independent on the choice of number of harmonics and number of time samples. Furthermore DNN is independent on the number of tones in the harmonic balance simulation. This flexibility of DNN is a clear progress over the existing behavioral neural models whose structure is dependent on the number of tones.

Although different in their implementations in circuit simulators, the two representations of DNN, i.e., circuit and HB representations, are numerically equivalent. The former representation is more convenient to implement and the latter is computationally more efficient.

4.2.5 Discussion

The proposed DNN automatically achieves a model reduction effect since the DNN order n can be chosen to be much less than the order of the original nonlinear circuit. By adjusting n , we can conveniently adjust the order of our model. Another factor in the DNN model is that the number of hidden neurons in f_{ANN} represents the extent of nonlinearity between dynamic inputs and dynamic outputs. By adjusting the number of hidden neurons, we can conveniently adjust the degree of nonlinearity needed in the DNN model. Such convenient adjustments of order and nonlinearity in DNN make the model creation much easier than conventional equivalent circuit based approaches where manual trial-and-error may be needed to create/adjust the equivalent circuit topology and the nonlinear equation terms in it.

In the DNN formulation (4.3) which is based on the representation of (4.2), the signal-flow from input to output resembles differentiation, hence the model can be referred to as Differential DNN (DDNN). An alternative formulation is an Integral DNN (IDNN) approach [156], where the input-output relationship is re-organized as,

$$y(t) = f(y^{(n)}(t), y^{(n-1)}(t), \dots, y^{(1)}(t), u^{(n)}(t), u^{(n-1)}(t), \dots, u(t)) \quad (4.17)$$

Here the signal-flow from input to output resembles integration, as such the model is called Integral DNN (IDNN).

These two different formulations, i.e., DDNN and IDNN, are theoretically equivalent and are complementary formats of DNN. They together form the DNN family. These two advanced techniques can represent the nonlinear RF/microwave circuit behavior and can be used as models for high-level circuit and system design.

4.3 Demonstration Examples

4.3.1 Example A: DNN Modeling of Amplifier

This example shows the modeling of nonlinear effects of an amplifier using the DNN technique. The amplifier internally has 9 NPN transistors modeled by *Agilent-ADS* nonlinear models Q34, Q37, and HP AT 41411 [128] shown in Figure 4.5.

We train our DNN to learn the input-output dynamics of the amplifier. We choose a hybrid 2-port formulation with $u=[v_{IN}, i_{OUT}]^T$ as input, and $y = [i_{IN}, v_{OUT}]^T$ as output. The DNN model includes,

$$i_{IN}^{(n)}(t) = f_{ANN1}(i_{IN}^{(n-1)}(t), i_{IN}^{(n-2)}(t), \dots, i_{IN}(t), v_{IN}^{(n)}(t), v_{IN}^{(n-1)}(t), \dots, v_{IN}(t)) \quad (4.18)$$

$$v_{OUT}^{(n)}(t) = f_{ANN2}(v_{OUT}^{(n-1)}(t), v_{OUT}^{(n-2)}(t), \dots, v_{OUT}(t), v_{IN}^{(n)}(t), v_{IN}^{(n-1)}(t), \dots, v_{IN}(t), i_{OUT}^{(n)}(t), i_{OUT}^{(n-1)}(t), \dots, i_{OUT}(t)) \quad (4.19)$$

This input-output definition allows the model to be able to interact with external connections with other nonlinear circuits in a system level simulation.

The training data for the amplifier is gathered by exciting the circuit with a set of frequencies (0.95 ~ 1.35GHz, step-size 0.05GHz), powers (-30 ~ -14 dBm, step-size 2 dBm), and load impedances (35 ~ 65 Ohms, step-size 10 Ohms). In initial training, Fourier Transform sampling frequencies ranged from 47.5 to 67.5GHz. Final training is done with optimization over harmonic balance such that modeled harmonics match original harmonics. We trained the model in multiple ways using different number of hidden neurons and orders (n) of the model as shown in Table 4.1.

Figure 4.5. Amplifier circuit to be represented by a DNN model.

Table 4.1. Amplifier: DNN accuracy from different training. Testing error for DNN with different number of hidden neurons (40, 50, 60) and different number of orders ($n = 2, 3, 4$) are computed. This table shows the results for different number of hidden neurons when $n = 3$, and the results (with the highest accuracy) for different number of orders.

No. of Hidden Neurons in Training ($n=3$)	Testing		Order n in Training	Testing	
	Error for Time Domain Data	Error for Spectrum Data		Error for Time Domain Data	Error for Spectrum Data
40	4.2E-3	2.7E-3	2	5.3E-3	4.3E-3
50	2.9E-3	1.8E-3	3	2.9E-3	1.8E-3
60	3.6E-3	2.3E-3	4	1.5E-2	9.9E-3

Testing is performed by comparing our DNN model with the original amplifier in ADS, with different set of signals never used in training, i.e., different test frequencies (0.975 ~ 1.325GHz, step-size 0.05 GHz), powers (-29 ~ -15 dBm, step-size 2 dBm) and loads (40, 50, 60 Ohms). The model is compared with the original circuit in both time and frequency domains, and excellent agreement is achieved. Figure 4.6 shows examples of spectrum comparisons. An additional comparison between our DNN model and the original amplifier is made using the 1-dB compression point. For example, at the excitation frequency 1.175 GHz, the 1-dB compression point is -35.6 dBm for the DNN model agreeing well with its original value of -35.0 dBm from the original amplifier.

We also applied envelope transient analysis to the DNN amplifier model using the ADS envelope simulator. The model was driven with a 1.15 GHz carrier and modulated by a $\pi/4$ – DQPSK signal at 48.6 kbits/s. The result of the simulation is illustrated in Figure 4.7, showing two cases of power spectral regrowth at the DNN output, (a) when the amplifier model operates at 1-dB compression point, and (b) when the amplifier model operates at 10-dB compression point.

To further demonstrate that the DNN model represents circuit internal behavior independent of external signals, we show a different use of the proposed technique for this amplifier. We use exactly the same formulation of the amplifier DNN model to handle 2-tone harmonic balance effects. To further add to the challenge of this modeling task, we perform the training of the DNN using 1-tone data and 1-tone formulation of training (optimization). After training is finished, we will use the model for 2-tone simulation.

Figure 4.6. Amplifier output: Spectrum comparison between DNN ($\square$), and ADS solution of original circuit ($\square$) at load = 50 Ω . Good agreement is achieved even though such data was never used in training.

(a)

Figure 4.7. (a) Envelope transient analysis results (output power spectrum) for DNN amplifier model with $\pi/4$ – DQPSK modulation, when the amplifier model operates at 1-dB compression point.

(b)

Figure 4.7. (b) Envelope transient analysis results (output power spectrum) for DNN amplifier model with $\pi/4$ – DQPSK modulation, when the amplifier model operates at 10-dB compression point.

This ability of the DNN demonstrates progress over existing behavioral based neural models where the model structure has to be different for different number of tones. The proposed DNN achieves uniform format regardless of the number of tones.

For this demonstration, the training data for the amplifier is gathered by exciting the circuit with several patterns of input signal $v_{IN}(t)$: fundamental frequencies (0.2 GHz, 0.22 GHz), powers at the fourth and the fifth harmonics (-24 ~ -20 dBm, step-size 2 dBm), and the total number of harmonics considered with harmonic balance simulation is 20. Testing is performed by comparing our model with original amplifier, with two-tone signal never used in training. For the first tone, fundamental frequency is 0.84 GHz, powers (-23 dBm, -21 dBm). For the second tone, fundamental frequency is 1.05 GHz, powers (-23 dBm, -21 dBm). The number of harmonics in the HB simulation for each tone is 4 leading to a total number of 20 harmonics and intermodulated frequencies in the output signal. The 2-tone solution from the DNN model is compared with the ADS solution of the original amplifier in both frequency and time domains, and excellent agreement is achieved as shown in Figures 4.8 and 4.9, respectively. We also computed the third-order intercept point (IP3). For example, when the two tone input powers are set to -23 dBm, the IP3 computed from our DNN model is 2.24 dBm, which is a good estimation of the original IP3 of 2.38 dBm from the original amplifier.

This example demonstrates that the same DNN structure can be used for single- or multi-tone harmonic balance simulations providing simplicity and flexibility in implementation, model development, and model usage over the existing neural network methods.

Figure 4.8. Amplifier 2-tone simulation result from DNN, which is trained under 1-tone formulation: Spectrum comparison between DNN ($\square$) and ADS solution of original circuit ($\square$). Good agreement is achieved even though such 2-tone data was never used in training.

Figure 4.9. Amplifier 2-tone simulation result from DNN: Time-domain comparison between DNN (—) and ADS solution of original circuit (o). Good agreement is achieved even though such data was never used in training.

4.3.2 Example B: Mixer DNN Modeling

This example illustrates DNN modeling of a mixer. The circuit internally is a Gilbert cell with 14 NPN transistors in ADS [128] shown in Figure 4.10. The dynamic input and output of the model is defined in hybrid form as $u=[v_{RF}, v_{LO}, i_{IF}]^T$ and $y = [i_{RF}, v_{IF}]^T$. The DNN model includes,

$$i_{RF}^{(n)}(t) = f_{ANN1}(i_{RF}^{(n-1)}(t), i_{RF}^{(n-2)}(t), \dots, i_{RF}(t), v_{RF}^{(n)}(t), v_{RF}^{(n-1)}(t), \dots, v_{RF}(t)) \quad (4.20)$$

$$v_{IF}^{(n)}(t) = f_{ANN2}(v_{IF}^{(n-1)}(t), v_{IF}^{(n-2)}(t), \dots, v_{IF}(t), v_{RF}^{(n)}(t), v_{RF}^{(n-1)}(t), \dots, v_{RF}(t), v_{LO}^{(n)}(t), v_{LO}^{(n-1)}(t), \dots, v_{LO}(t), i_{IF}^{(n)}(t), i_{IF}^{(n-1)}(t), \dots, i_{IF}(t)) \quad (4.21)$$

The training data is gathered as follows. RF input frequency and power level changed from 11.7 to 12.1GHz with step-size 0.05GHz and from -45 dBm to -35 dBm with step-size 2 dBm, respectively. LO signal is fixed at 10.75 GHz and 10dBm. The load is perturbed by 10% at every harmonic in order to let the model learn the load effects. The DNN is trained with a different number of hidden neurons and orders (n) as shown in Table 4.2.

Testing is done in ADS using input frequencies (11.725 ~ 12.075GHz, step-size 0.05GHz) and power levels (-44, -42, -40, -38, -36 dBm). The agreement between model and ADS is achieved in time and frequency domains even though those test information was never seen in training. Figure 4.11 illustrates examples of test in the time domain.

4.3.3 Example C: Nonlinear Simulation of DBS Receiver System

To further confirm the validity of the proposed DNN, we also trained a DNN model representing another amplifier (gain stage amplifier) using similar way as that in Example A,

Figure 4.10. Mixer equivalent circuit to be represented by a DNN model.

Table 4.2. Mixer: DNN accuracy from different training. Testing error for DNN with different number of hidden neurons (45, 55, 65) and different number of orders ($n = 2, 3, 4$) are computed. This table shows the results for different number of hidden neurons when $n = 4$, and the results (with the highest accuracy) for different number of orders.

No. of Hidden Neurons in Training ($n=4$)	Testing Error for Time Domain Data	Testing Error for Spectrum Data
45	8.7E-4	6.7E-4
55	4.6E-4	2.0E-4
65	6.5E-4	4.6E-4

Order n in Training	Testing Error for Time Domain Data	Testing Error for Spectrum Data
2	2.7E-3	1.9E-3
3	1.4E-3	8.6E-4
4	4.6E-4	2.0E-4

Figure 4.11. Mixer V_{IF} output: Time-domain comparison between DNN (—) and ADS solution of original circuit (o). Good agreement is achieved even though such data was never used in training.

and combined the three trained DNNs of mixer and amplifiers into a DBS receiver sub-system [157], where the amplifier trained in Example A is used as the output stage. The overall DBS system is shown in Figure 4.12.

We have incorporated the DNN models of the amplifiers and mixer into harmonic balance simulation in two ways. The first way is to use the circuit representation of DNNs as described in Figure 4.4(a) incorporated into ADS software. This is achieved by constructing the equivalent circuit in ADS using capacitors, controlled sources and algebraic expressions representing f_{ANN} neural network function. The second way is to program the HB representation of DNN model of Figure 4.4(b) for amplifiers and mixer according to (4.16). The overall DBS system output solved by the efficient HB representation of DNNs match completely with that solved using circuit representation of DNNs in ADS, confirming the consistency between the two representations of DNN as shown in Figure 4.13(a). Next we compare ADS harmonic balance simulation with original DBS system in Figure 4.12(a) with that using DNN models of amplifiers and mixer in Figure 4.12(b). The overall DBS system solution using DNNs matches that of the original system as shown in Figure 4.13(b), even though these obviously distorted signals were never used in training of any of the DNNs.

We also performed Monte-Carlo analysis of the original and the DNN based DBS systems under random sets of RF input frequencies and power levels. The statistics from the DNN based system simulation, shown in Figure 4.14, matches that from the original system.

The CPU for 1000 analyses of the DBS system using original circuits, using circuit representation of DNNs, and using HB representation of DNNs are 6.52, 3.94 and 0.81 hours, respectively, showing efficiency of the DNN based system simulation.

Figure 4.12. DBS receiver sub-system: (a) connected by the original detailed equivalent circuit in ADS, (b) connected by our DNNs.

(a)

Figure 4.13 (a). DBS system output: Comparison between system solutions using HB representation of DNN models (—), and circuit representation of DNN models (x). The solutions from the two representations of DNN are in good agreement of each other.

(b)

Figure 4.13 (b). DBS system output: Comparison between system solutions using DNN models (—), and ADS simulation of original system (o). Good agreement is achieved even though these nonlinear solutions were never used in training.

Figure 4.14. Histogram of power gain of DBS system for 1000 Monte Carlo simulations with random input frequency and amplitude.

A further comparison is made between the proposed dynamic neural network, the conventional static neural network approach, and the conventional behavioral modeling approach (non-neural network approach). We trained three static neural networks using the static I-Q (current-charge) model of [64] to learn the two amplifiers and the mixer, and incorporated these models into ADS using *NeuroADS* [158]. The overall DBS system simulation using the static neural models was performed in ADS. As expected, such static models, while suitable for intrinsic FET modeling, are not accurate enough for amplifier and mixers even though the model incorporates charge information. The overall error in the output signal of the DBS system is 6.1% relative to original detailed system simulation.

For the case of conventional behavioral modeling, we constructed three behavioral models to represent the two amplifiers and the mixer. The behavioral models were obtained in two ways, one way is to use the data based behavioral model [128], and another way is to use optimization to optimize the behavioral model parameters in [128] to best match the behavior of the original amplifiers and the mixer. An overall DBS system simulation with the best behavioral models was used. As expected, the behavioral models run extremely fast, and provide only an approximate solution. Table 4.3 provides a summary of model test error for the two amplifiers and one mixer through different methods. Table 4.4 provides comparisons of computation speed and accuracy with the different methods for the DBS system simulation. It is observed that the proposed DNN (i.e., dynamic neural network) approach provides the best overall performance being much faster than original system simulation and much more accurate than both the conventional behavioral modeling approach, and the static neural network approach.

4.4 Summary

This chapter presented a neural network method for modeling nonlinear microwave devices or circuits and its applications for high-level system simulation. The model is derived in continuous time-domain dynamic format and can be developed from input-output data without having to rely on internal details of the circuits. A novel training scheme allows the training of DNN to learn from either time or frequency domain input-output information. After being trained, the proposed model can be conveniently incorporated into existing simulators. Compared to existing neural based methods, the DNN retains or enhances the neural modeling speed and accuracy capabilities, and provides additional flexibility in handling diverse needs of nonlinear microwave simulation, e.g., time and frequency domain applications, single- and multi-tone simulations. The technique allows further realizing the flexibility of neural based approaches in nonlinear microwave modeling, simulation and optimization.

Table 4.3. DBS system component models: Testing

error comparison (for spectrum data) between conventional behavioral model, static neural model, and DNNs.

Techniques Components	Conventional Behavioral Model	Static I-Q Neural Model	Proposed DNN Model
Mixer	3.4%	3.2%	0.02%
Gain stage Amplifier	1.2%	1.9%	0.09%
Output stage Amplifier	7.7%	2.9%	0.16%

Table 4.4. DBS-receiver sub-system: Accuracy and computation speed comparisons between system simulation using

conventional behavioral model, static neural model, DNNs, and detailed original circuit. It is observed that the proposed DNN with

HB representation provides the best overall performance being much faster than original system simulation and much more accurate than both the conventional behavioral modeling approach, and the static neural network approach.

Techniques Comparisons	DBS system simulation using Conventional Behavioral Model	DBS system simulation using Static I-Q Neural Model	DBS system simulation using HB Representation of DNNs	DBS system simulation using Circuit Representation of DNNs	DBS system simulation using Detailed Original Circuit
Test Error for Spectrum Data	10.3%	6.1%	0.21%	0.21 %	0.0% (reference for comparison)
CPU time for 1000 Monte- Carlo Analysis	0.18 hours	0.26 hours	0.81 hours	3.94 hours	6.52 hours

CHAPTER 5

Neural Based Microwave Modeling and Design using Advanced Model Extrapolation

Further progress of neural based nonlinear microwave device/circuit modeling is made by a new technique presented in this chapter, i.e., an advanced neural model extrapolation technique [7]. It enables neural based nonlinear microwave device/circuit models to be robustly used in iterative computational loops, e.g., HB simulation, involving neural model inputs as iterative variables. A new process is incorporated in training to formulate a set of base points to represent a regular or irregular training region. An adaptive base point selection method is developed to identify the most significant subset of base points upon any given value of model input. Combining quadratic approximation with the information of the model at these base points including both the input/output behavior and its derivatives, this technique is able to reliably extrapolate the performance of the model from training range to a much larger region, substantially improving the convergence of the iterative computational loops involving the trained neural models.

5.1 Introduction

Neural network based nonlinear microwave device or circuit modeling, which is generally followed by neural-model-based circuit or system design, has been successfully applied to numerous high-frequency CAD problems. The progress of ANN-based RF/microwave CAD depends on innovative research activities that can further strengthen

it in terms of accuracy and speed, and make it even more attractive for practical applications. This chapter presents a novel algorithm for obtaining an improved convergence of iterative computational loops involving neural models. This is achieved by using advanced model extrapolation to improve the performance of a trained neural model beyond its training range.

A neural network model, after being trained for a particular range of data, is very good at representing the original problem within the training region [2]. However, outside this region, the accuracy of the model deteriorates very rapidly due to saturation of the activation functions in the hidden layer of neural network structure [2]. This creates limitations for use of neural models in iterative computational loops such as optimization and HB simulation where the range of the iterative variables may need to be much larger than the neural model training range. This is an important issue for microwave design involving physical/geometrical design parameters and nonlinear circuit simulation. The poor performance of conventional neural model outside the training range may mislead the iterative process into slow convergence or even divergence.

For the first time, the task of using microwave neural models far beyond their training range is addressed. A new process is proposed in training to formulate a set of base points to represent a regular or irregular training region. An adaptive base point selection method is developed to identify the most significant subset of base points upon any given value of model input. Combining quadratic approximation [159][160] with the information of the model at these base points including both the input/output behavior and its derivatives, the proposed technique is able to reliably extrapolate the performance of the model from training range to a much larger region.

5.2 Neural Based Model Extrapolation Technique

Within this chapter, we use $x = [x_1 \ x_2 \ \dots \ x_{N_x}]$ and $y = [y_1 \ y_2 \ \dots \ y_{N_y}]$ to represent the input and output vectors of feedforward neural network parts of a microwave modeling problem, where N_x and N_y are the number of inputs and the number of outputs, respectively. Let $\mathcal{R}_t$ be defined as the training region, which can be represented by boundary value of each variable for training region with rectangular boundary, or approximated by the whole set of training data for training region with arbitrary boundary. Let the neural network model be defined as,

$$y = f_{ANN}(x), \quad x \in \mathcal{R}_t \quad (5.1)$$

An essential part of a neural network is the activation function in hidden neurons. Many hidden neurons allow the neural model to represent multidimensional input-output behavior accurately [2]. However outside the training region, i.e., $x \notin \mathcal{R}_t$, the activation functions saturate rapidly and the trained neural model carries little information about the original problem. Here, we present an algorithm for improving the performance of a trained model over an extended region.

5.2.1 Base Points for Extrapolation

Model extrapolation is performed based on the available information for the model, defined by a set of base points inside training region. Let $\mathcal{R}_b$ be the total set of basis points. Let T_i be i^{th} training point, $i=1, 2, \dots, N_t$, where N_t is the total number of training samples in $\mathcal{R}_t$. Let $\tilde{B}_i$ be i^{th} base point, $i=1, 2, \dots, N_b$, where N_b is the total number of base points in $\mathcal{R}_b$. In order to speed up the model extrapolation, we propose a data pre-process to extract the

effective set of base points for extrapolation, $\mathfrak{R}_b$, from $\mathfrak{R}_t$, i.e., $\mathfrak{R}_b$ spans the same space as $\mathfrak{R}_t$ and $N_b \ll N_t$. The proposed data process is carried out at the end of the training process and is formulated such that irregular training region with arbitrary boundary can be accommodated.

Let $\tilde{S}_j$ be the number of intervals for model input x_j , representing user specified resolution for extrapolation base points. Let $x_{j,min}$, $x_{j,max}$ be the minimum and maximum values of model input x_j , respectively. The proposed technique will firstly define $N_r = \tilde{S}_1 * \tilde{S}_2 * \dots * \tilde{S}_{N_x}$ grid-subregions in the x space. The center of each subregion $\tilde{C}_i = [\tilde{C}_{i,1}, \tilde{C}_{i,2}, \dots, \tilde{C}_{i,N_x}]$ can be uniquely identified by its index $\tilde{I}_i = [\tilde{I}_{i,1}, \tilde{I}_{i,2}, \dots, \tilde{I}_{i,N_x}]$, $i=1, 2, \dots, N_r$, where,

$$\tilde{C}_{i,j} = \frac{x_{j,max} - x_{j,min}}{\tilde{S}_j} \cdot (\tilde{I}_{i,j} + 0.5), \quad j=1, 2, \dots, N_x \quad (5.2)$$

Secondly, $\mathfrak{R}_t$ is mapped into those subregions. If a training point belongs to subregion i , we call that the subregion i is occupied. The centers of occupied subregions will be defined as base points for extrapolation. The set of such base points are computed by initializing,

$$\mathfrak{R}_b = \emptyset \quad (5.3)$$

Then for $i=1, 2, \dots, N_r$, we update $\mathfrak{R}_b$ as,

$$\mathfrak{R}_b = \mathfrak{R}_b \cup \tilde{C}_k \quad (5.4)$$

where k is such that for all $j, j=1, 2, \dots, N_x$,

$$\tilde{C}_{k,j} - \frac{x_{j,max} - x_{j,min}}{2 \cdot \tilde{S}_j} < T_{i,j} < \tilde{C}_{k,j} + \frac{x_{j,max} - x_{j,min}}{2 \cdot \tilde{S}_j} \quad (5.5)$$

The whole process for extracting the effective set of base points for extrapolation is illustrated in Figure 5.1.

Figure 5.1. Processing $\mathcal{R}_t$ to obtain the effective set of base points for extrapolation, $\mathcal{R}_b$.

This process is carried out at the end of the training process.

5.2.2 Computation of Model Extrapolation

Given inputs x , the proposed technique will firstly search $\mathfrak{R}_b$ to find several base points closest to x , i.e., $\tilde{B}_i$, $i \in \tilde{P}$, where $\tilde{P}$ is the index set of those base points. Let N_p be a user defined parameter representing the number of points allowed in $\tilde{P}$. The distance $\tilde{d}$ between x and the base points in $\mathfrak{R}_b$ is defined as,

$$\tilde{d}_i = \|x - \tilde{B}_i\|, \quad i = 1, 2, \dots, N_p \quad (5.6)$$

where $\tilde{d}_i < \tilde{d}_j$, for $i \in \tilde{P}$ and $j \notin \tilde{P}$.

Then a smooth quadratic function will be used to best match the behavior of those N_p points, including both the input/output behavior and their derivatives [161]. Here we propose a weighting matrix $\tilde{W}$ to regulate the amount of influence of the base points. The quadratic approximation using $\tilde{W}$ can be formulated as,

$$\tilde{W} \cdot \tilde{A} \cdot \tilde{V} = \tilde{W} \cdot \tilde{b} \quad (5.7)$$

where $\tilde{V}$ represents the parameters in the quadratic function, and $\tilde{A}, \tilde{b}$ represent the input/output information and the derivatives, provided by the adjoint neural network as described in Chapter 3, at the base points $\tilde{B}_i$, $i \in \tilde{P}$.

In order to solve (5.7), the least square method [160] is applied as

$$\tilde{V} = ((\tilde{W} \tilde{A})^T \tilde{W} \tilde{A})^{-1} \cdot (\tilde{W} \tilde{A})^T \cdot \tilde{W} \cdot \tilde{b} \quad (5.8)$$

The computation of the proposed model extrapolation is illustrated in Figure 5.2.

Figure 5.2. Flow-chart of the proposed model extrapolation. This process is done during the use of the trained neural models.

5.3 Demonstration Examples

5.3.1 Example A: Neural Based Design Solution Space Analysis of Coupled Transmission Lines

This example illustrates that the proposed technique can be applied to improve the performance of the neural model involved in optimization, e.g., design solution space analysis [3][4] which is very useful for synthesis of components and for making trade-off decisions during the early design stages of systems. A basic step is to use optimization to find the inputs x of the model from given specifications on y .

In this example, the neural model,

$$L_{12} = f_{ANN}(w_1, w_2, s, h, \epsilon_r, frequency) \quad (5.9)$$

with $x = [w_1, w_2, s, h, \epsilon_r, frequency]^T$ as ANN input, and $y = [L_{12}]^T$ as ANN output, is used to model the cross-sectional mutual inductance of coupled transmission lines shown in Figure 5.3., for analysis of high speed VLSI interconnects [89], where w_1 , w_2 , s , h , and ϵ_r are conductor widths, spacing, substrate thickness and dielectric constants, respectively. We will use optimization to find corresponding reasonable w_1 , w_2 , s , h , and ϵ_r for a given design budget of mutual inductance L_{12} . In this optimization, the history of x often goes beyond the training range, even though the initial values are inside the training region as shown in Figure 5.4. Without the proposed model extrapolation, optimization is misled to a wrong solution, because the model is not reliable outside the training range. With the proposed technique, the optimization reached the correct solution. Table 5.1 shows the comparison of the convergence range between non-extrapolated and extrapolated models for different given design budget of mutual inductance L_{12} , where the effect of the

proposed technique is clearly shown. This example demonstrates that the proposed method allows the trained neural models to be used more reliably in circuit optimization.

Figure 5.3. Coupled transmission lines for analysis of high speed VLSI interconnects, where w_1 , w_2 , s , h , and ϵ_r are conductor widths, spacing between coupled interconnects, substrate thickness, and dielectric constant, respectively. A neural network model is to be trained for this transmission line, and the model is to be used to demonstrate the proposed advanced neural model extrapolation technique.

Figure 5.4. The optimization trajectory of design parameters w_1 and s in coupled transmission lines example. As observed, w_1 and s can go beyond the training range during the optimization process, even though their initial values are inside the training region.

Table 5.1. Convergence range
(relative distance from solution) of non-extrapolated and extrapolated
neural model for coupled transmission line example

L_{12}	ANN without Extrapolation	ANN with Extrapolation
11.7 nH	[-90%, 90%]	[-500%, 500%]
101.0 nH	[-80%, 80%]	[-400%, 400%]
306.0 nH	[-100%, 100%]	[-500%, 500%]

5.3.2 Example B: Neural Based Behavior Modeling and Simulation of Power Amplifiers

This example illustrates that the proposed technique can be applied to dynamic neural network (DNN) models, as described in Chapter 4, to improve the performance of the model involved in HB simulation for RF/microwave circuit and system design.

In this example, the DNN model,

$$v_{out} = f_{ANN}(v_{out}^{(1)}, v_{out}^{(2)}, v_{out}^{(3)}, v_{in}, v_{in}^{(1)}, v_{in}^{(2)}, v_{in}^{(3)}) \quad (5.10)$$

with $x = [v_{out}^{(1)}, v_{out}^{(2)}, v_{out}^{(3)}, v_{in}, v_{in}^{(1)}, v_{in}^{(2)}, v_{in}^{(3)}]^T$ as ANN input, and $y = [v_{out}]^T$ as ANN output, is used to model the unidirectional input-output dynamic relations of a power amplifier

[5][6]. The training data for the amplifier is gathered by exciting the circuit with a set of frequencies (0.8 ~ 1.2 GHz, step-size 0.1 GHz), powers (-5 ~ 15 dBm, step-size 2 dBm). Even though the frequencies and powers are grid distributed, they are not directly the ANN input variables. The actual ANN inputs are $\mathbf{x} = [v_{out}^{(1)}, v_{out}^{(2)}, v_{out}^{(3)}, v_{in}, v_{in}^{(1)}, v_{in}^{(2)}, v_{in}^{(3)}]^T$, which are dependent on each other, resulting in an irregular training region $\mathfrak{R}_t$, as shown by solid lines in Figure 5.5. With the proposed training process, the effective set of base points for extrapolation can be obtained as shown by circles in Figure 5.5.

Figure 5.5. Training region (-) and effective set of base points for extrapolation (o) shown in subspace of v_{in} and $v_{out}^{(1)}$, for the DNN power amplifier example.

HB has its own schemes of creating initial values for the solution process and also requests the outputs of the models from HB-supplied inputs. In this process, it is often possible that the current/voltage inputs to the model, decided by the HB algorithm, are

outside the ANN training region. Figure 5.6. shows an actual HB iterative history requiring the voltage inputs $v_{out}^{(1)}$, $v_{out}^{(2)}$, and $v_{out}^{(3)}$ of DNN model far beyond the training region, where a reliable model performance is needed to improve the HB convergence. Table 5.2 shows the comparison of the convergence range between non-extrapolated and extrapolated models for different input powers. The proposed technique enables the HB simulation to converge over a larger range than the non-extrapolated model.

Figure 5.6. HB simulation of power amplifier: solid lines represent the training region and the circles represent the HB simulation history of $v_{out}^{(1)}$, $v_{out}^{(2)}$, and $v_{out}^{(3)}$. As shown in the figure, the x values of the model will go far beyond the training region during HB, necessitating the use of the proposed technique.

Table 5.2. Convergence range

(relative distance from solution) of non-extrapolated and extrapolated
DNN model for modeling the power amplifier input-output relationship

Pin	DNN without Extrapolation	DNN with Extrapolation
-4 dBm	[-180%, 180%]	[-400%, 400%]
6 dBm	[-60%, 60%]	[-300%, 300%]
14 dBm	[-10%, 10%]	[-150%, 150%]

5.3.3 Example C: Neural Based Bidirectional Behavior Modeling and Simulation of Power Amplifiers

To further confirm the validity of the proposed technique, we apply it to a bidirectional DNN formulation to improve the performance of the model involved in HB simulation.

In this example, the DNN model,

$$i_{out} = f_{ANN}(i_{out}^{(1)}, i_{out}^{(2)}, i_{out}^{(3)}, v_{in}, v_{in}^{(1)}, v_{in}^{(2)}, v_{in}^{(3)}, v_{out}, v_{out}^{(1)}, v_{out}^{(2)}, v_{out}^{(3)}) \quad (5.11)$$

with $x = [i_{out}^{(1)}, i_{out}^{(2)}, i_{out}^{(3)}, v_{in}, v_{in}^{(1)}, v_{in}^{(2)}, v_{in}^{(3)}, v_{out}, v_{out}^{(1)}, v_{out}^{(2)}, v_{out}^{(3)}]^T$ as ANN input, and $y = [i_{out}]^T$ as ANN output, is used to model the bidirectional input-output dynamic relations of the same power amplifier as Example B. The training data for the amplifier is gathered by

exciting the circuit with a set of frequencies (0.8 ~ 1.2 GHz, step-size 0.1 GHz), V_{in} powers (-5 ~ 15 dBm, step-size 2 dBm), and V_{out} is sampled to cover a certain range for each harmonics, corresponding to a set of linear/nonlinear loads. Similar to Example B, the training region $\mathcal{R}_t$ for the actual input variables of the neural model is irregular. This will necessitate the proposed training process to extract the effective set of base points for extrapolation, $\mathcal{R}_b$. Testing is performed by exciting the amplifier circuit with a set of frequencies (0.85 ~ 1.15 GHz, step-size 0.1 GHz), V_{in} powers (-4 ~ 14 dBm, step-size 2 dBm), and connecting it with linear/nonlinear external load never seen in training.

With the proposed technique, the trained neural model can extend the behavior of the model from the training range to a much larger region, effectively improving the performance of the model, shown in Table 5.3.

5.4 Summary

An advanced neural model extrapolation technique has been proposed for improving the performance of a trained neural based nonlinear microwave device or circuit model beyond its training range. The proposed technique enables neural models to be robustly used in iterative computational loops, e.g., optimization and HB simulation, involving neural model inputs as iterative variables. Compared with standard neural based methods (i.e., without extrapolation), the proposed technique improves neural based microwave optimization and makes nonlinear circuit design significantly more robust. The effectiveness of the proposed algorithm has been demonstrated by examples of neural based design solution space analysis of coupled transmission lines and neural based behavior modeling and simulation of power amplifiers.

Table 5.3. Convergence range

(relative distance from solution) of non-extrapolated and extrapolated

DNN model for modeling the power amplifier two port input-output relationship

Pin (dBm)	External Load	DNN without Extrapolation	DNN with Extrapolation
-4	Linear	[-150%, 150%]	[-300%, 300%]
-4	Nonlinear	[-120%, 120%]	[-280%, 280%]
6	Linear	[-55%, 55%]	[-220%, 220%]
6	Nonlinear	[-50%, 50%]	[-200%, 200%]
14	Linear	[-10%, 10%]	[-150%, 150%]
14	Nonlinear	[-6%, 6%]	[-105%, 105%]

CHAPTER 6

Conclusions and Future Research

6.1 Conclusions

Rapid progress in the RF and microwave electronics industry over the last decade has led to dramatic increase in circuit complexity and size. Conventional design procedures became increasingly difficult with ever-increasing circuit complexities coupled with tightened design tolerances [1]. Computer-aided design is essential for achieving performance and yield in high-frequency electronic circuits and systems. Modeling is a major bottleneck for efficient computer-aided design and optimization of RF/microwave components and circuits. Recently, neural network based CAD that advocates the use of accurate and fast neural models in place of computationally prohibitive theoretical models has gained recognition [1].

The thesis has presented state-of-the-art research in the neural-network-based RF/microwave CAD area. The central objectives of the thesis are efficient and accurate nonlinear microwave device and circuit modeling.

It is envisaged that meeting such key objectives through focused and methodical research in this area could enable its success in terms of accuracy, cost-effectiveness, speed and viability when employed in practical CAD applications. Specifically, in view of the above-mentioned objectives and vision, the following contributions have been made through this thesis work.

An adjoint neural network (ADJNN) algorithm [3][4] has been proposed for neural based device/circuit modeling, and sensitivity analysis for generic types of microwave neural models including knowledge based models. This method provided continuous and differentiable models with analytically consistent derivatives from raw information present in the original training data. A novel and elegant first- and second-order sensitivity analysis scheme has been developed allowing the training of neural models to learn not only input-output relationships in a microwave component but also its derivatives, which is very useful in simultaneous DC/small-signal/large-signal device or circuit modeling.

A dynamic neural network (DNN) method [5][6] for modeling nonlinear microwave devices and circuits used for high-level simulation has been proposed. The model was derived in an effective format, i.e., continuous time-domain dynamic format. The model can be developed from input-output data without having to rely on internal details of the device or circuit. A novel training scheme has been developed allowing the training of DNN to learn from either time or frequency domain input-output information. After being trained, the proposed model can be conveniently incorporated into existing simulators. The DNN retains or enhances the advantages of learning, speed, and accuracy as in existing neural network techniques; and provides additional advantages of being theoretically elegant and practically suitable for diverse needs of nonlinear microwave simulation, e.g., standardized implementation in simulators, suitability for both time and frequency domain applications, and multi-tone simulations.

The issue of using neural-based nonlinear microwave device and circuit models far outside their training range has been directly addressed in the advanced neural model extrapolation

technique [7]. This technique enables neural based nonlinear microwave device and circuit models to be robustly used in iterative computational loops involving neural model inputs as iterative variables. A new process has been incorporated in training to formulate a set of base points to represent a regular or irregular training region. An adaptive base point selection method has been developed to identify the most significant subset of base points upon any given value of model input. This method was combined with quadratic extrapolation utilizing neural network outputs and their derivatives. It improves neural based microwave optimization and makes nonlinear circuit design significantly more robust over that using standard neural based methods.

In order to accelerate the practical use of the algorithms proposed in this thesis, a generalized implementation is necessary. An object-oriented computer program embedding ADJNN, DNN and the advanced neural model extrapolation algorithms has been developed in C++. The program has been used in deriving the results in this thesis and incorporated into a trial version of the *NeuroModeler* software [8].

The research works in this thesis, i.e., ADJNN technique, DNN technique, and the advanced neural model extrapolation technique, are important contributions to further realizing the flexibility of neural based approaches in nonlinear microwave modeling, simulation and optimization.

6.2 Future Directions

Artificial neural networks represent one of the most recent trends in RF and microwave computer aided design, and neural-network-based CAD approaches including those

presented in this thesis aim to achieve high levels of speed and precision, in an attempt to meet the challenges posed by the next generation of high-frequency design. This area combines artificial intelligence concepts with state-of-the-art CAD technologies creating many opportunities for technical discovery and industrial applications at various stages of high frequency CAD including modeling, simulation and design.

Combining the learning and the generalization capabilities of artificial neural networks with existing RF and microwave engineering knowledge continues to be a strategic area of ANN based research. An interesting topic would be the idea of incorporating existing dynamic knowledge of the original nonlinear circuit into the present DNN modeling framework. The objective is to utilize available knowledge maximally to develop hybrid circuit-DNN model architectures that can attain highest levels of model accuracies especially when limited training data is available. As described in Chapter 4, the proposed DNN algorithm directly utilizes the external input and output information of the nonlinear circuit for training without referring to any details inside the circuit. In reality, any available knowledge of the original circuit should help to achieve higher accuracy with less effort, e.g., less training data and better extrapolation property. How to embed such dynamic knowledge into DNN model and develop a suitable training scheme will be a new challenge topic of research.

As shown in Chapter 5, the DNN model with extrapolation capability is more robust than the non-extrapolated DNN when used in iterative computational loops involving neural model inputs as iterative variables, e.g., HB simulation. This motivates the investigation for different methods of extrapolation to obtain a globally robust DNN model, which allows the starting point of the iterative computational loops involving DNN models to be random. In

this way, the DNN model will be more suitable for nonlinear microwave circuit modeling and will be of greater practical importance in future.

The proposed DNN modeling technique has been demonstrated to be efficient for modeling the nonlinear microwave circuit, e.g., amplifier and mixer as shown in Chapter 4. An interesting direction is exploiting the potential of DNN to directly model the sub-system involving many nonlinear microwave circuits, e.g., DBS receiver sub-system. In this way, many efforts for developing DNN models for individual nonlinear circuits in a sub-system can be avoided, and the large system-level simulation and design can be further speeded up with sufficient accuracy.

Since the DNN applications in this thesis are mainly steady state analysis of the nonlinear RF/microwave circuit, another interesting direction is to expand the DNN technique for time domain transient analysis and applications. This will lead to a new training scheme, including a new dynamic adjoint neural network method to best match the transient training data. It will also involve the stability analysis of the model. Such analysis will further lead to the formulation of a constrained training in order to obtain more stable DNN models.

Last but not least, another significant milestone in the area is to incorporate the RF/microwave-oriented neural network modeling algorithms and techniques into readily usable ANN-software tools. These tools enable RF/microwave designers to quickly build neural models for their high-frequency devices, circuits and systems, and feedback from designers can further stimulate advanced research in the modeling area. Most of the commercially available RF/microwave network simulators do have provision for linking externally developed component models including neural network models. A key

requirement is to develop plug-in software tools that can allow convenient insertion of neural models into commercial simulator environments for carrying out circuit- and system-level CAD. These activities have tremendous potential for propagating neural based design and optimization into newer arenas of electronics CAD.

In conclusion, artificial neural networks with their unique qualities of accuracy, flexibility and speed, continue to be one of the most attractive and powerful vehicles at the forefront of RF and microwave computer aided design.

APPENDIX A

Using Adjoint Neural Network Model and Dynamic Neural Network Model in *Agilent-ADS* for Circuit/System Simulation and Design

The adjoint neural network model and dynamic neural network model presented in this thesis can be used by *Agilent-ADS* [142] through an ADS plug-in module called *NeuroADS* [168]. Along with this thesis work, more software codes have been developed enabling the *NeuroADS* to conveniently implement large-signal nonlinear device or circuit neural models into *Agilent-ADS*. Neural based nonlinear models can then be used together with existing CAD tool's library models to perform simulation, optimization and statistical design of high-level circuits and systems.

The implementation of a neural based model into ADS requires the development of a set of user-defined model following ADS template [142]. The three main steps in developing a user-defined model are:

1. Defining the parameters that the user will interface with the model from the ADS schematic.
2. Defining the circuit symbol and number of pins for interfacing with ADS simulators.
3. Development of C code.

The first two steps consist of developing the application extension language (AEL) code to interface the model with ADS. AEL provides the coupling of the model's parameters

and pins in the schematic design to the simulator. The C code is used to define the component's response to its parameter configuration, simulation controls and pin voltages. In the follow sections, we will describe the three main steps for developing a user-defined model in detail.

A.1 The Interface between Neural Model and ADS

Here we use a practical example to describe the interface between neural model and ADS. Consider two-port neural based nonlinear current source (I_{ds}) in Example 3 of Chapter 3, which has the schematic and parameters shown in Figure A.1.

Figure A.1. Schematic and model parameters of two-port neural based nonlinear current source (I_{ds}) in Example C of Chapter 3, implemented in ADS using user-defined model.

ADS will use such interface to provide the coupling between neural model and simulator, i.e., given the model's parameters and voltages on the pins by ADS, the internal neural model will supply the corresponding response back to simulator.

A.2 General Function Blocks of C Code

ADS has several types of simulators in which a model can be used. The linear simulator is used for emulating the small-signal response versus frequency of a component. Typical outputs include S-parameters, stability factor and maximum available gain. A small-signal or a large-signal model can be used within the linear simulator. The nonlinear simulator (harmonic balance) is used for the large-signal response of a component at different power excitations. The transient simulator is used to model the response of a component versus time. The harmonic balance and transient simulators require a large-signal model that accurately represents the behavior of a component.

The neural network based user-defined model code contains three main functions that correspond to linearized sub-network, nonlinear sub-network and neural network sub-network, which follows the standardized format of representing nonlinear component in typical nonlinear circuit simulators. The linear function formulates the admittance matrix for each node. The nonlinear function formulates the nonlinear current and nonlinear charge for each port. Finally, the neural network function computes the nonlinear currents and charges, and provides them to the nonlinear sub-network. It also computes the derivatives of the currents and charges w.r.t each voltage source and provides the values to the linearized sub-network. In this way, the neural network based nonlinear modes is able to be used together with existing CAD tool's library models to perform DC/small-signal/large-signal simulation, optimization and statistical design of high-level circuits and systems. Figure A.2. shows the block diagram of the relationships between the three main functions in neural network based user-defined model and the ADS circuit simulator.

Figure A.2. Block diagram of the relationships between the three main functions in neural network based user-defined model and the ADS circuit simulator.

A.2.1 ADJNN in ADS

ADJNN has been implemented into ADS in the form of nonlinear voltage controlled current sources and nonlinear voltage controlled charge sources.

Given the port voltages as the inputs, the original neural model is activated to produce the nonlinear currents (i_{ANN}) or charges (q_{ANN}). At the same time, the adjoint neural model produces the derivatives of the currents or charges w.r.t each port voltage, i.e., di_{ANN}/dv or dq_{ANN}/dv . Those values will be used for computing the nonlinear sub-network and the linearized sub-network respectively, as shown in Figure A.2.

A.2.2 DNN in ADS

The DNN model, i.e., Equation (4.3), is in a standardized format for typical nonlinear circuit simulators. For example, the left-hand-side of the equation provides the charge (Q) or the capacitor part, and the right-hand-side provides the current (I) part. Such standard representation of DNN enables the convenient incorporation of trained model into ADS. The right-hand-side of Equation (4.3) has been implemented into ADS in the form of multi-port nonlinear voltage controlled current source involving neural model as the control function. The overall DNN model in ADS is constructed by connecting such nonlinear current source with some standard capacitors, where those capacitors correspond to the left-hand-side of the Equation (4.3).

Given the port voltages, this nonlinear voltage controlled current source produces the port currents and their derivatives w.r.t each port voltage, corresponding to the formulation defined in the right-hand-side of Equation (4.3). The resulting values will be used for computing the nonlinear sub-network and the linearized sub-network respectively.

In the nonlinear voltage controlled current source, the evaluation and differentiation of f_{ANN} part of the DNN model, as shown in Equation (4.3), is accomplished by the original neural model and its adjoint neural model respectively, which is illustrated in Figure A.3.

Figure A.3. The evaluation and differentiation of f_{ANN} part of the DNN model is accomplished by the original neural model and its adjoint neural model respectively. The resulting values will be used for computing the nonlinear sub-network and the linearized sub-network respectively.

Bibliography

- [1] Q.J. Zhang and K.C. Gupta, *Neural Networks for RF and Microwave Design*, Norwood, MA: Artech House, 2000.
- [2] S. Haykin, *Neural Networks, A Comprehensive Foundation*, New Jersey: Prentice Hall, 1994.
- [3] J.J Xu, M.C.E. Yagoub and Q.J. Zhang, "Exact adjoint sensitivity for neural based microwave modeling and design," *IEEE MTT-S Int. Microwave Symp. Digest*, (Phoenix, AZ), pp. 1015-1018, May 2001.
- [4] J.J Xu, M.C.E. Yagoub and Q.J. Zhang, "Exact adjoint sensitivity for neural based microwave modeling and design," *IEEE Trans. Microwave Theory Tech.*, vol. 51, pp. 226-237, 2003.
- [5] J.J. Xu, M.C.E. Yagoub, R. Ding and Q.J. Zhang, "Neural based dynamic modeling of nonlinear microwave circuits," *IEEE MTT-S Int. Microwave Symp. Digest*, (Seattle, WA), pp. 1101-1104, June 2002.
- [6] J.J. Xu, M.C.E. Yagoub, R. Ding and Q.J. Zhang, "Neural based dynamic modeling of nonlinear microwave circuits," *IEEE Trans. Microwave Theory Tech.*, vol. 50, pp. 2769-2780, 2002.
- [7] J.J. Xu, M.C.E. Yagoub, R. Ding and Q.J. Zhang, "Robust Neural Based Microwave Modeling and Design using Advanced Model Extrapolation," *IEEE MTT-S Int. Microwave Symp. Digest*, (Fort Worth, Texas), pp. 1549-1552, June 2004.
- [8] *NeuroModeler Version 1.2*, Prof. Q.J. Zhang, Department of Electronics, Carleton University, 1125 Colonel By Drive, Ottawa, Ontario, K1S 5B6, Canada.

- [9] B.S. Cooper, "Selected Applications of Neural Networks in Telecommunication Systems," *Australian Telecommunication Research*, Vol. 28, pp. 9-29, 1994.
- [10] G. Cheron, J.P. Draye, M. Bourgeios and G. Libert, "A dynamic neural network identification of electromyography and arm trajectory relationship during complex movements," *IEEE Trans. Biomedical Engineering*, vol. 43, pp. 552-558, 1996.
- [11] J.R. Noriega and H. Wang, "A direct adaptive neural-network control for unknown nonlinear systems and its application," *IEEE Trans. Neural Networks*, vol. 9, pp. 27-34, 1998.
- [12] B. Hussain and M.R. Kabuka, "A novel feature recognition neural network and its application to character recognition," *IEEE Trans. Pattern Anal. Machine Intelligence*, vol. 16, pp. 98 -106, 1994.
- [13] A. Waibel, T. Hanazawa, G Hinton, K. Shikano and K.J Lang, "Phoneme recognition using time-delay neural networks," *IEEE Trans. Acoustics Speech Signal Processing*, vol. 37, pp. 328-339, 1989.
- [14] J.F. Nunmaker Jr. and R.H. Sprague Jr., "Applications of Neural Networks in Manufacturing," *Proc. Int. Conf. System Sciences*, (Weilea, HI), pp. 447-53, January, 1996.
- [15] M.H. Bakr, J.W. Bandler, M.A. Ismail, J.E. Rayas-Sánchez and Q.J. Zhang "Neural space mapping optimization for EM-based design," *IEEE Trans. Microwave Theory Tech.*, vol. 48, pp. 2307-2315, 2000.
- [16] A. Veluswami, M.S. Nakhla and Q.J. Zhang, "The application of neural networks to EM-based simulation and optimization of interconnects in high-speed VLSI circuits," *IEEE Trans. Microwave Theory Tech.*, vol. 45, pp. 712-723, 1997.

- [17] P.M. Watson and K. C. Gupta, "EM-ANN models for microstrip vias and interconnects in dataset circuit," *IEEE Trans. Microwave Theory Tech.*, vol. 44, pp. 2495-2503, 1996.
- [18] A. Veluswami, Q.J. Zhang and M.S. Nakhla, "A neural network model for propagation delays in systems with high-speed VLSI interconnect networks," *Proc. IEEE Custom Integrated Circuits Conf.*, (Santa Clara, CA), pp. 387-390, May 1995.
- [19] K. Shirakawa, M. Shimizu, N. Okubo and Y. Daido, "Structural determination of multilayered large-signal neural network HEMT model," *IEEE Trans. Microwave Theory Tech.*, vol. 46, pp. 1367-1375, 1998.
- [20] F. Wang and Q.J. Zhang, "Knowledge based neural models for microwave design," *IEEE Trans. Microwave Theory Tech.*, vol. 45, pp. 2333-2343, 1997.
- [21] R. Biernacki, J.W. Bandler, J. Song and Q.J. Zhang, "Efficient quadratic approximation for statistical design," *IEEE Trans. Circuits Syst.*, vol. CAS-36, pp. 1449-1454, 1989.
- [22] P.B.L. Meijer, "Fast and smooth highly nonlinear multidimensional table models for device modeling," *IEEE Trans. Circuits Syst.*, vol. 37, pp. 335-346, 1990.
- [23] J. Bandler, M. Ismail, J. Rayas-Sanchez and Q. 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, July 1999.
- [24] J.W. Bandler, M.A. Ismail, J.E. Rayas-Sanchez and Q.J. Zhang, "Neuromodeling of microwave circuits exploiting space-mapping technology," *IEEE Trans. Microwave Theory Tech.*, vol. 47, pp. 2417-2427, 1999.

- [25] P.M. Watson, K.C. Gupta and R.L. Mahajan, "Development of knowledge based artificial neural network models for microwave components," *IEEE MTT-S Int. Microwave Symp. Digest*, (Baltimore, MD), pp. 9-12, June 1998.
- [26] P.M. Watson, K.C. Gupta and R.L. Mahajan, "Applications of knowledge-based artificial neural network modeling to microwave components," *Int. J. RF and Microwave CAE*, vol. 9, pp. 254-260, 1999.
- [27] K.C. Gupta, "EM-ANN models for microwave and millimeter-wave components," *IEEE MTT-S Int. Microwave Symp. Workshop on Applications of ANN to Microwave Design*, (Denver, CO), pp. 17-47, June 1997.
- [28] P. Watson and K.C. Gupta, "EM-ANN models for via interconnects in microstrip circuits," *IEEE MTT-S Int. Microwave Symp. Digest*, (San Francisco, CA), pp. 1819-1822, June 1996.
- [29] P.M. Watson and K.C. Gupta, "Design and optimization of CPW circuits using EM-ANN models for CPW components," *IEEE Trans. Microwave Theory Tech.*, vol. 45, pp. 2515-2523, 1997.
- [30] P. Watson, G. Creech and K. Gupta, "Knowledge based EM-ANN models for the design of wide bandwidth CPW patch/slot antennas," *IEEE AP-S Int. Symp. Digest*, (Orlando, FL), pp. 2588-2591, July 1999.
- [31] G.L. Creech, B.J. Paul, C.D. Lesniak, T.J. Jenkins and M.C. Calcaterra "Artificial neural networks for fast and accurate EM-CAD of microwave circuits," *IEEE Trans. Microwave Theory Tech.*, vol. 45, pp. 794-802, 1997.

- [32] G.L. Creech, B. Paul, C. Lesniak, T. Jenkins, R. Lee and M. Calcaterra, "Artificial neural networks for accurate microwave CAD applications," *IEEE MTT-S Int. Microwave Symp. Digest*, (San Francisco, CA), pp. 733-736, June 1996.
- [33] Y. Harkouss, J. Rousset, H. Chehade, E. Ngoya, D. Barataud and J.P. Teyssier, "The use of artificial neural networks in nonlinear microwave devices and circuits modeling: An application to telecommunication system design," *Int. J. RF Microwave CAE*, vol. 9, pp. 198-215, 1999.
- [34] A.H. Zaabab, Q.J. Zhang and M. Nakhla, "Analysis and optimization of microwave circuits & devices using neural network models," *IEEE MTT-S Int. Microwave Symp. Digest*, (San Diego, CA), pp. 393-396, May 1994.
- [35] G. Kothapalli, "Artificial neural networks as aids in circuit design," *Microelectronics J.*, vol. 26, pp. 569-578, 1995.
- [36] V.B. Litovski, J.I. Radjenovic, Z.M. Mrcarica and S.L. Milenkovic, "MOS transistor modeling using neural network," *Elect. Lett.*, vol. 28, pp. 1766-1768, 1992.
- [37] G.L. Creech and J.M. Zurada, "Neural network modeling of GaAs IC material and MESFET device characteristics," *Int. J. RF and Microwave CAE*, vol. 9, pp. 241-253, 1999.
- [38] J. Rousset, Y. Harkouss, J.M. Collantes and M. Campovecchio, "An accurate neural network model of FET intermodulation and power analysis," *Proc. European Microwave Conf.*, (Prague, Czech Republic), pp. 16-19, September 1996.
- [39] A.H. Zaabab, Q.J. Zhang and M. Nakhla, "A neural network modeling approach to circuit optimization and statistical design," *IEEE Trans. Microwave Theory Tech.*, vol. 43, pp. 1349-1358, 1995.

- [40] J.A. Garcia, A.T. Puente, A.M. Sanchez, I. Santamaria, M. Lazaro, C.J. Pantaleon and J.C. Pedro, "Modeling MESFETs and HEMTs intermodulation distortion behavior using a generalized radial basis function network," *Int. J. RF Microwave CAE*, vol. 9, pp. 261-276, 1999.
- [41] S. Goasguen, S.M. Hammadi and S.M. El-Ghazaly, "A global modeling approach using artificial neural network," *IEEE MTT-S Int. Microwave Symp. Digest*, (Anaheim, CA), pp. 153-156, June 1999.
- [42] G.L. Creech, "Neural networks for the design and fabrication of integrated circuits," *IEEE MTT-S Int. Microwave Symp. Workshop on Applications of ANN to Microwave Design*, (Denver, CO), pp. 67-86, June 1997.
- [43] V.B. Litovski, J.I. Radjenovic, Z.M. Mrcarica and S.L. Milenkovic, "MOS transistor modeling using neural network," *Elect. Lett.*, vol. 28, pp. 1766-1768, 1992.
- [44] 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. European Microwave Conf.*, (Amsterdam, Netherlands), pp. 306-311, October 1998.
- [45] 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, 1993.
- [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), pp. 1054-1057, July 2000.

- [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 Tech. Conf.*, (San Jose, CA), pp. 1134-1137, May 1997.
- [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. Microwave Symp. Digest*, (Boston, MA), pp. 879-882, June 2000.
- [49] P. Burrascano, M. Dionigi, C. Fancelli and M. Mongiardo, "A neural network model for CAD and optimization of microwave filters," *IEEE MTT-S Int. Microwave Symp. Digest*, (Baltimore, MD), pp. 13-16, June 1998.
- [50] G. Fedi, A. Gaggelli, S. Manetti and G. Pelosi, "Direct-coupled cavity filters design using a hybrid feedforward neural network - finite elements procedure," *Int. J. RF Microwave CAE*, vol. 9, pp. 287-296, 1999.
- [51] S. Bila, Y. Harkouss, M. Ibrahim, J. Rousset, E. N'Goya, D. Billargeat, S. Verdeyme, M. Auborg and P. Guillon, "An accurate wavelet neural-network-based model for electromagnetic optimization of microwave circuits," *Int. J. RF Microwave CAE*, vol. 9, pp. 297-306, 1999.
- [52] S. Verdeyme, D. Billargeat, S. Bila, S. Moraud, H. Blondeaux, M. Aubourg and P. Guillon, "Finite element CAD for microwave filters," *Proc. European Microwave Conf. Workshop*, (Amsterdam, Netherlands), pp. 12-22, October 1998.
- [53] P. Burrascano, S. Fiori and M. Mongiardo, "A review of artificial neural networks applications in microwave computer-aided design," *Int. J. RF Microwave CAE*, vol. 9, pp. 158-174, 1999.

- [54] M. Vai, S. Wu, B. Li and S. Prasad, "Creating neural network based microwave circuit models for analysis and synthesis," *Proc. Asia Pacific Microwave Conf.*, (Hong Kong), pp. 853-856, December 1997.
- [55] Y. Fang, M. Yagoub, F. Wang and Q.J. Zhang, "A new macromodeling approach for nonlinear microwave circuits based on recurrent neural networks," *IEEE Trans. Microwave Theory Tech.*, vol. 48, pp. 2335-2344, 2000.
- [56] C. Christodoulou, A. El Zooghby and M. Georgiopoulos, "Neural network processing for adaptive array antennas," *IEEE AP-S Int. Symp. Digest*, (Orlando, FL), pp. 2584-2587, July 1999.
- [57] R. Mishra and A. Patnaik, "Neurospectral analysis of coaxial fed rectangular patch antenna," *IEEE AP-S Int. Symp. Digest*, (Salt Lake City, UT), pp. 1062-1065, July 2000.
- [58] A.H. El Zooghby, C.G. Christodoulou and M. Georgiopoulos, "Neural network-based adaptive beamforming for one- and two-dimensional antenna arrays," *IEEE Trans. Antennas Propagat.*, vol. 46, pp. 1891-1893, 1998.
- [59] E. Charpentier and J.J. Laurin, "An implementation of a direction-finding antenna for mobile communications using a neural network," *IEEE Trans. Antennas Propagat.*, vol. 47, pp. 1152-1159, 1999.
- [60] S. Sagiroglu, K. Guney and M. Erler, "Calculation of bandwidth for electrically thin and thick rectangular microstrip antennas with the use of multilayered perceptrons," *Int. J. RF Microwave CAE*, vol. 9, pp. 277-286, 1999.

- [61] S. El-Khamy, M. Aboul-Dahab and K. Hijjah, "Sidelobes reduction and steerable nulling of antenna arrays using neural networks," *IEEE APS Int. Symp. Digest*, (Orlando, FL), pp. 2600-2603, July 1999.
- [62] G. Castaldi, V. Pierro and I.M. Pinto, "Neural net aided fault diagnostics of large antenna arrays," *IEEE APS Int. Symp. Digest*, (Orlando, FL), pp. 2608-2611, July 1999.
- [63] S. Goasguen, S.M. Hammadi and S.M. El-Ghazaly, "A global modeling approach using artificial neural network," *IEEE MTT-S Int. Microwave Symp. Digest*, (Anaheim, CA), pp. 153-156, June 1999.
- [64] A.H. Zaabab, Q.J. Zhang and M.S. Nakhla, "Neural network modeling approach to circuit optimization and statistical design," *IEEE Trans. Microwave Theory Tech.*, vol. 43, pp. 1349-1358, 1995.
- [65] P.M. Watson, C. Cho and K.C. Gupta "Electromagnetic-artificial neural network model for synthesis of physical dimensions for multilayer asymmetric coupled transmission structures," *Int. J. RF and Microwave CAE*, Special Issue on Applications of ANN to RF and Microwave Design, vol. 9, pp. 175-186, 1999.
- [66] M. Vai and S. Prasad, "Neural networks in microwave circuit design – beyond black box models," *Int. J. RF and Microwave CAE*, Special Issue on Applications of ANN to RF and Microwave Design, vol. 9, pp. 187-197, 1999.
- [67] F. Wang, V.K. Devabhaktuni, C. Xi and Q.J. Zhang, "Neural network structures and training algorithms for RF and microwave applications," *Int. J. RF Microwave CAE*, vol. 9, pp. 216-240, 1999.

- [68] D.E. Rumelhart, G.E. Hinton and R.J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing*, vol. I, D.E. Rumelhart and J.L. McClelland, Eds., Cambridge, MA: MIT Press, 1986, pp. 318-362.
- [69] W.H. Press, B.P. Flannery, S.A. Teukolsky and W.T. Vetterling, *Numerical Recipes: The Art of Scientific Computing*, Cambridge, MA: Cambridge University Press, 1986.
- [70] T.R. Cuthbert Jr., "Quasi-Newton methods and constraints," in *Optimization Using Personal Computers*, New York, NY: John Wiley & Sons, 1987, pp. 233-314.
- [71] A.J. Shepherd, "Second-order optimization methods," *Second-Order Methods for Neural Networks*, Berlin, NY: Springer-Verlag, 1997, pp. 43-72.
- [72] J.A. Nelder and R. Mead, "A simplex method for function minimization," *Computer Journal*, vol. 7, pp. 308-313, 1965.
- [73] S. Kirkpatrick, C.D. Gelatt and M.P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671-680, 1983.
- [74] J.C.F. Pujol and R. Poli, "Evolving neural networks using a dual representation with a combined crossover operator," *Proc. IEEE Intl. Conf. Evol. Comp.*, (Anchorage, Alaska), pp. 416-421, May 1998.
- [75] K. Hornik, M. Stinchcombe and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, pp. 359-366, 1989.
- [76] V.K. Devabhaktuni, M.C.E. Yagoub, Y. Fang, J. Xu and Q.J. Zhang, "Neural networks for microwave modeling: Model development issues and nonlinear modeling techniques," *Int. J. RF and Microwave CAE*, vol. 11, pp. 4-21, 2001.

- [77] 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, pp. 630-645, 1997.
- [78] R. Reed, "Pruning algorithms - A survey," *IEEE Trans. Neural Networks*, vol. 4, pp. 740-747, 1993.
- [79] A. Krzyzak and T. Linder, "Radial basis function networks and complexity regularization in function learning," *IEEE Trans. Neural Networks*, vol. 9, pp. 247-256, 1998.
- [80] S. Tamura and M. Tateishi, "Capabilities of a four-layered feedforward neural network: Four layer versus three", *IEEE Trans. Neural Networks*. Vol. 8, pp.251-255, 1997.
- [81] F. Girosi, *From Statistics to Neural Networks: Theory and Pattern Recognition Applications*, chapter Regularization Theory, Radial Basis Functions and Networks, pp. 166-187. Springer-Verlag, Berlin, NY, 1992.
- [82] M.J.D. Powell, *Algorithms for Approximation*, chapter Radial Basis Functions for Multivariate Interpolation: A Review, pp. 143-167. Oxford University Press, Oxford, UK, 1987.
- [83] J. Park and I.W. Sandberg, "Universal approximation using radial-basis-function networks," *Neural Computation*, vol. 5, pp. 305-316, 1993.
- [84] J. Park and I.W. Sandberg, "Approximation and radial-basis-function networks," *Neural Computation*, vol. 3, pp. 246-257, 1991.
- [85] A. Krzyzak, T. Linder and G. Lugosi, "Nonparametric estimation and classification using radial basis function nets and empirical risk minimization," *IEEE Trans. Neural Networks*, vol. 7, pp. 475-487, 1996.

- [86] Q.H. Zhang, "Using wavelet network in nonparametric estimation," *IEEE Trans. Neural Networks*, vol. 8, pp. 227-236, 1997.
- [87] Q.H. Zhang and A. Benvensite, "Wavelet networks," *IEEE Trans. Neural Networks*, vol. 3, pp. 889-898, 1992.
- [88] B.R. Bakshi and G. Stephanopoulos, "Wave-net: a multiresolution, hierarchical neural network with localized learning," *America Institute of Chemical Eng. Journal*, vol. 39, pp. 57-81, 1993.
- [89] Q.J. Zhang, F. Wang and M.S. Nakhla, "Optimization of high-speed VLSI interconnects: A review," *Int. J. Microwave Millimeter-Wave CAE*, vol. 7, pp. 83-107, 1997.
- [90] Q. J. Zhang, K. C. Gupta and V. K. Devabhaktuni, "Artificial neural networks for RF and microwave design: From theory to practice," *IEEE Trans. Microwave Theory Tech.*, vol. 51, pp. 1339 – 1350, 2003.
- [91] D.G. Luenberger, *Linear and Nonlinear Programming*, Addison-Wesley, Reading, Massachusetts, 1989.
- [92] D.B. Parker, "Optimal algorithms for adaptive network: second order back propagation, second order direct propagation and second order hebbian learning," *Proc. IEEE First Intl. Conf. Neural Networks*, (San Diego, CA), pp. 593-600, June 1987.
- [93] R.L. Watrous, "Learning algorithms for connectionist networks: applied gradient methods of nonlinear optimization," *Proc. IEEE First Intl. Conf. Neural Networks*, (San Diego, CA), pp. 619-627, June 1987.

- [94] R. Battiti, "First and second-order methods for learning: between steepest descent and Newton's method," *Neural Computation*, vol. 4, pp. 141-166, 1992.
- [95] J.S.R. Jang, C.T. Sun and E. Mizutani, "Derivative-based optimization," In *Neuro-Fuzzy and Soft Computing, A Computational Approach to Learning and Machine Intelligence*, Upper Saddle River, NJ: Prentice Hall, 1997, pp. 129-172.
- [96] N. Baba, Y. Mogami, M. Kohzaki, Y. Shiraishi and Y. Yoshida, "A hybrid algorithm for finding the global minimum of error function of neural networks and its applications," *Neural Networks*, vol. 7, pp. 1253-1265, 1994.
- [97] M.B. Steer, J.W. Bandler and C.M. Snowden, "Computer-aided design of RF and microwave circuits and systems," *IEEE Trans. Microwave Theory Tech.*, vol. 50, pp. 996-1005, 2002.
- [98] C.M. Snowden, *Semiconductor Device Modeling*, Stevenage, U.K.: Peregrinus, 1989.
- [99] K. Lehovec and R. Zuleeg, "Voltage-current characteristics of GaAs JFET's in the hot electron range," *Solid State Electron.*, vol. 13, pp. 1415-1426, 1970.
- [100] P.H. Ladbrooke, *MMIC Design: GaAs FET's and HEMT's*, Norwood, MA: Artech House, 1989.
- [101] Q. Li and R.W. Dutton, "Numerical small-signal AC modeling of deep-level-trap related frequency dependent output conductance and capacitance for GaAs MESFET's on semi-insulating substrates," *IEEE Trans. Electron Devices*, vol. 38, pp. 1285-1288, 1991.
- [102] M.A. Khatibzadeh and R.J. Trew, "A large-signal analytical model for the GaAs MESFET," *IEEE Trans. Microwave Theory Tech.*, vol. 36, pp. 231-238, 1988.

- [103] *MINIMOS-NT v.2.0*, Institute for Microelectronics, Technical University Vienna, Vienna, Austria.
- [104] R.J. Trew, "MESFET models for microwave CAD applications," *Int. J. Microwave Millimeter-Wave Computer-Aided Eng.*, vol. 1, pp. 143–58, 1991.
- [105] A. Materka and T. Kacprzak, "Computer calculation of large-signal GaAs FET amplifier characteristics," *IEEE Trans. Microwave Theory Tech.*, vol. 33, pp. 129–135, 1985.
- [106] M.B. Steer, J.W. Bandler and C.M. Snowden, "Computer-aided design of RF and microwave circuits and systems," *IEEE Trans. Microwave Theory Tech.*, vol. 50, pp. 996–1005, 2002.
- [107] C.M. Snowden, *Semiconductor Device Modeling*, Stevenage, U.K.: Peregrinus, 1989.
- [108] D. Schreurs, J. Verspecht, S. Vandenberghe, G. Carchon, K. van der Zanden and B. Nauwelaers, "Easy and accurate empirical transistor model parameter estimation from vectorial large-signal measurements," *IEEE Int. Microwave Symposium Digest*, 1999.
- [109] D. Root, S. Fan and J. Meyer, "Technology independent large signal quasi-static FET models by direct construction from automatically characterized device data," *Proc. European Microwave Conf.*, (Stuttgart, Germany), pp. 927–932, September 1991.
- [110] J.M. Golio, *The RF and Microwave Handbook*. Boca Raton, FL: CRC Press, 2001.
- [111] W.R. Curtice, "A MESFET model for use in the design of GaAs integrated circuits," *IEEE Trans. Microwave Theory Tech.*, vol. 28, pp. 448–456, 1980.

- [112] T. Kacprzak and A. Materka, "Compact DC model of GaAs FETs for large-signal computer calculation," *IEEE J. Solid-State Circuits*, vol. Scull, pp. 211–213, 1983.
- [113] H. Statz, R. Newman, I. W. Smith, R. A. Pucel and H. A. Haus, "GaAs FET device and circuit simulation in SPICE," *IEEE Trans. Electron De-vices*, vol. ED-34, pp. 160–169, 1987.
- [114] J.M. Golio, *Microwave MESFET's & HEMT's*, Norwood, MA: Artech House, 1991.
- [115] L. Angelov, H. Zirath and N. Rorsman, "New empirical nonlinear model for HEMT and MESFET and devices," *IEEE Trans. Microwave Theory Tech.*, vol. 40, pp. 2258–2266, 1992.
- [116] V.I. Cojocaru and T.J. Brazil, "A scalable general-purpose model for microwave FET's including the DC/Ac dispersion effects," *IEEE Trans. Microwave Theory Tech.*, vol. 12, pp. 2248–2255, 1997.
- [117] H.K. Gummel and Poon, "An integral charge-control model of bipolar transistors," *Bell Syst. Tech. J.*, vol. 49, pp. 827–852, 1970.
- [118] C.M. Snowden, "Nonlinear modeling of power FET's and HBT's," *Int. J. Microwave and Millimeter-Wave Computer-Aided Eng.*, vol. 6, pp. 219–33, 1996.
- [119] H. Zaabab, Q.J. Zhang and M.S. Nakhla, "Device and circuit level modeling using neural networks with faster training based on network sparsity", *IEEE Trans. Microwave Theory Tech.*, vol. 45, pp. 1696-1704, 1997.
- [120] 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. Microwave Theory Tech.*, vol. 50, pp. 2315-2319, 2002.

- [121] OSA90 v.3.0, Optimization Systems Associates, P.O. Box 8083, Dundas, Canada, L9H 5E7, now Agilent EEsof, 1400 Fountaingrove Parkway, Santa Rosa, CA 95403.
- [122] K. Shirakawa, M. Shimizu, N. Okubo and Y. Daido, "Structural determination of multilayered large signal neural network HEMT model," *IEEE Trans. Microwave Theory Tech.*, vol. 46, pp. 1367-1375, 1998.
- [123] K. Shirakawa, M. Shimizu, N. Okubo and Y. Daido, "A large signal characterization of an HEMT using a multilayered neural network," *IEEE Trans. Microwave Theory Tech.*, vol. 45, pp. 1630-1633, 1997.
- [124] B. Davis, C. White, M.A. Reece, M.E. Bayne, W.L. Thompson, N.L. Richardson and L. Walker, "Dynamically configurable pHEMT model using neural networks for CAD," *IEEE MTT-S Int. Microwave Symp. Digest*, (Philadelphia, PA), pp. 177-180, 2003.
- [125] P. Vismuller, *RF Design Guide, Systems, Circuits and Equations*, Norwood, MA: Artech House, 1995.
- [126] T.R. Turlington, *Behavioral Modeling of Nonlinear RF and Microwave Devices*, Boston, MA: Artech House, 2000.
- [127] J. Verspecht, F. Verbeyst, M.V. Bossche and P. Van Esch, "System level simulation benefits from frequency domain behavioral models of mixers and amplifiers," *Proc. European Microwave Conf.*, (Munich, Germany), pp. 29-32, October 1999.
- [128] *ADS-Advanced Design System Version 2002*, Agilent Technologies, Santa Rosa, CA, 2002.

- [129] H. Ku, M.D. McKinley and J.S. Kenney, "Extraction of accurate behavioral models for power amplifiers with memory effects using two-tone measurements," *IEEE MTT-S Int. Microwave Symp. Digest*, (Seattle, WA), pp. 139-142, 2002.
- [130] P. Reig, N. LeGallou, J.M. Nebus and E. Ngoya, "Accurate RF and microwave system level modeling of wideband nonlinear circuits," *IEEE MTT-S Int. Microwave Symp. Digest*, (Boston, MA), pp. 79-82, 2000.
- [131] N. LeGallou, E. Ngoya, H. Buret, D. Barataud and J.M. Nebus, "An improved behavioral modeling technique for high power amplifiers with memory," *IEEE MTT-S Int. Microwave Symp. Digest*, (Phoenix, AZ), pp. 983-986, 2001.
- [132] P. Tuinenga, "Models rush in where simulators fear to tread: extending the baseband-equivalent method," *IEEE Int. Behavioral Modeling and Simulation Conf.*, (Santa Rosa, CA), pp. 32-40, 2002.
- [133] J. Verspecht, D. Schreurs, A. Barel and B. Nauwelaers, "Black box modeling of hard nonlinear behavioral in the frequency domain," *IEEE MTT-S Int. Microwave Symp. Digest*, (San Francisco, CA), pp. 1735-1738, 1996.
- [134] J. Verspecht and P.V. Esch, "Accurately characterizing hard nonlinear behavioral of microwave components with the nonlinear network measurement system: introducing 'nonlinear scattering functions'," *Proc. International Workshop on Integrated Nonlinear Microwave and Millimeterwave Circuits*, (Duisburg, Germany), pp. 17-26, October 1998.
- [135] R. Boyle, B.M. Cohn, D.O. Pederson and J.E. Soloman, "Macromodeling of integrated operational amplifiers," *IEEE J. Solid-State Circuits*, vol. 9, pp. 353-363, 1974.

- [136] G. Casinovi and A. Sangiovanni-Vincentelli, "A macromodeling algorithm for analogue circuits", *IEEE Trans. Computer-Aided Design*, vol. 10, pp. 150-160, 1991.
- [137] P.K. Gunupudi and M.S. Nakhla, "Model-reduction of nonlinear circuits using Krylov-space techniques," *Proc. IEEE Int. Design Automation Conf.*, (New Orleans, Louisiana), pp. 13-16, 1999.
- [138] P.K. Gunupudi and M.S. Nakhla, "Nonlinear Circuit-Reduction of High-Speed Interconnect Networks using Congruent Transformation Techniques," *IEEE Trans. Advanced Packaging*, vol. 24, pp. 317-325, 2001.
- [139] V. Rizzoli, A. Neri, D. Masotti and A. Lipparin, "A new family of neural network-based bi-directional and dispersive behavioral models for nonlinear RF/Microwave subsystems," *Int. J. RF and Microwave CAE*, vol. 12, pp. 51-70, 2002.
- [140] T. Liu, S. Boumaiza and F.M. Ghannouchi, "Dynamic behavioral modeling of 3G power amplifiers using real-valued time-delay neural networks," *IEEE Trans. Microwave Theory Tech.*, vol. 52, pp. 1025-1033, 2004.
- [141] D. Schreurs, M. Myslinski and K. A. Remley, "RF Behavioral Modeling from Multisine Measurements: Influence of Excitation Type," *Proc. European Microwave Conf.*, Munich, Germany, pp. 1011-1014, October 2003.
- [142] D. Schreurs, N. Tufillaro, J. Wood, D. Usikov, L. Barford and D. Root, "Development of time-domain behavioral models for microwave devices and ICs from vectorial large-signal measurements and simulations," *Proc. European GaAs and related III-V compounds applications symp.*, (Paris, France), pp. 236-239, October 2000.

- [143] B.A. Pearlmutter, "Gradient calculations for dynamic recurrent neural networks: a survey," *IEEE Trans. Neural Networks*, vol. 6, pp. 1212-1228, 1995.
- [144] J.W. Bandler, Q.J. Zhang and R. Biernacki, "A unified theory for frequency-domain simulation and sensitivity analysis of linear and nonlinear circuits," *IEEE Trans. Microwave Theory Tech.*, vol. 36, pp. 1661-1669, 1988.
- [145] J.W. Bandler and S. Chen, "Circuit optimization: the state of the art," *IEEE Trans. Microwave Theory Tech.*, vol. 36, pp. 424-443, 1988.
- [146] J.W. Bandler, R.M. Biernacki, S.H. Chen, J. Song, S. Ye and Q.J. Zhang, "Analytically unified DC/small-signal/large-signal circuit design," *IEEE Trans. Microwave Theory Tech.*, vol. 39, pp. 1076-1082, 1991.
- [147] M. Vai and S. Prasad, "Neural networks in microwave circuit design – beyond black box models," *Int. J. RF and Microwave CAE*, Special Issue on Applications of ANN to RF and Microwave Design, vol. 9, pp. 187-197, 1999.
- [148] G. Antonini and A. Orlandi, "Gradient evaluation for neural-networks-based electromagnetic optimization procedures," *IEEE Trans. Microwave Theory Tech.*, vol. 48, pp. 874-876, 2000.
- [149] A. Djordjević, R.F. Harrington, T. Sarkar and M. Baždar, *Matrix Parameters for Multiconductor Transmission Lines: Software and User's Manual*, Boston, MA: Artech House, 1989.
- [150] A. Neri, C. Cecchetti and A. Lipparini, "Fast prediction of the performance of wireless links by simulation-trained neural networks," *IEEE MTT-S Int. Microwave Symp. Digest*, (Boston, MA), pp. 429-432, 2000.

- [151] J.J. Hopfield and T.W. Tank, "‘Neural’ computation of decisions in optimization problems," *Biological Cybernetics*, vol. 52, pp. 141-152, 1985.
- [152] T. Hrycyk, *Neurocontrol, Towards An Industrial Control Methodology*, New York: Wiley- Interscience, 1997.
- [153] J. Vlach and K. Singhal, *Computer Methods for Circuit Analysis and Design*. New York, NY: Van Nostrand Reinhold, 1993.
- [154] 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," *Proc. IEEE MTT-S Int. Microwave Symp.*, (Philadelphia, PA), pp. 165-168, June 2003.
- [155] K.S. Kundert, G.B. Sorkin and A. Sangiovanni-vincentelli, "Applying harmonic balance to almost-periodic circuits," *IEEE Trans. Microwave Theory Tech.*, vol. 36, pp. 366-378, 1988.
- [156] M. Deo, J.J. Xu and Q.J. Zhang, "A new formulation of dynamic neural network for modeling of nonlinear RF/Microwave circuits," *Proc. European Microwave Conf.*, (Munich, Germany), pp. 1019-1022, October 2003.
- [157] M.C.E. Yagoub and H. Baudrand, "Optimum design of nonlinear microwave circuits," *IEEE Trans. Microwave Theory Tech.*, vol. 42, pp. 779-786, 1994.
- [158] *NeuroADS*, Prof. Q.J. Zhang, Department of Electronics, Carleton University, 1125 Colonel By Drive, Ottawa, Canada, K1S 5B6.
- [159] R.M. Biernacki, J.W. Bandler, J. Song and Q.J. Zhang, "Efficient quadratic approximation for statistical design," *IEEE Trans. on Circuits Syst.*, vol. 36, pp. 1449-1454, 1989.

- [160] J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimisation*, Bristol, England: Adam Hilger, 1990.
- [161] C. Brezinski and M.R. Zaglia, *Extrapolation Methods: Theory and Practice*, New York, NY: North-Holland, 1991.