Neural Network Inverse Modeling and Applications to Microwave Filter Design

Humayun Kabir, *Student Member, IEEE*, Ying Wang, *Member, IEEE*, Ming Yu, *Senior Member, IEEE*, and Qi-Jun Zhang, *Fellow, IEEE*

***Abstract*—In this paper, systematic neural network modeling techniques are presented for microwave modeling and design using the concept of inverse modeling where the inputs to the inverse model are electrical parameters and outputs are geometrical parameters. Training the neural network inverse model directly may become difficult due to the nonuniqueness of the input–output relationship in the inverse model. We propose a new method to solve such a problem by detecting multivalued solutions in training data. The data containing multivalued solutions are divided into groups according to derivative information using a neural network forward model such that individual groups do not have the problem of multivalued solutions. Multiple inverse models are built based on divided data groups, and are then combined to form a complete model. A comprehensive modeling methodology is proposed, which includes direct inverse modeling, segmentation, derivative division, and model combining techniques. The methodology is applied to waveguide filter modeling and more accurate results are achieved compared to the direct neural network inverse modeling method. Full electromagnetic simulation and measurement results of Ku-band circular waveguide dual-mode pseudoelliptic bandpass filters are presented to demonstrate the efficiency of the proposed neural network inverse modeling methodology.**

***Index Terms*—Computer-aided design, inverse modeling, microwave filter modeling, neural networks.**

I. Introduction

In RECENT years, neural network techniques have been recognized as a powerful tool for microwave design and modeling problems. It has been applied to various microwave design applications [1], [2] such as vias and interconnects [3], embedded passives [4], coplanar waveguide components [5], transistor modeling [6]–[8], noise modeling [9], power-amplifier modeling [10], analysis of multilayer shielded microwave circuits [11], nonlinear microwave circuit optimization [12], etc. Neural networks have the ability to model multidimensional nonlinear relationships. The evaluation from input to output of a trained neural network model is also very fast. These features make neural networks a useful alternative for device modeling where a mathematical model is not available or repetitive electromagnetic (EM) simulation is required. Once a model is developed, it can be used over and over again. This avoids repetitive EM simulation where a simple change in the physical dimension requires a complete re-simulation of the EM structure.

A neural network trained to model original EM problems can be called the forward model where the model inputs are physical or geometrical parameters and outputs are electrical parameters. For design purposes, the information is often processed in the reverse direction in order to find the geometrical/physical parameters for given values of electrical parameters, which is called the inverse problem. There are two methods to solve the inverse problem, i.e., the optimization method and direct inverse modeling method. In the optimization method, the EM simulator or the forward model is evaluated repetitively in order to find the optimal solutions of the geometrical parameters that can lead to a good match between modeled and specified electrical parameters. An example of such an approach is [13]. This method of inverse modeling is also known as the synthesis method.

The formula for the inverse problem, i.e., compute the geometrical parameters from given electrical parameters, is difficult to find analytically. Therefore, the neural network becomes a logical choice since it can be trained to learn from the data of the inverse problem. We define the input neurons of a neural network to be the electrical parameters of the modeling problem and the output neurons as the geometrical parameters. Training data for the neural network inverse model can be obtained simply by swapping the input and output data used to train the forward model. This method is called the direct inverse modeling and an example of this approach is [14]. Once training is completed, the direct inverse model can provide inverse solutions immediately unlike the optimization method where repetitive forward model evaluations are required. Therefore, the direct inverse model is faster than the optimization method using either the EM or the neural network forward model. A similar concept has been utilized in the neural inverse space mapping (NISM) technique where the inverse of the mapping from the fine to the coarse model parameter spaces is exploited in a space-mapping algorithm [15].

Though the neural network inverse model can provide the solution faster than the optimization method, it often encounters the problem of nonuniqueness in the input–output (IO) relationship. It also causes difficulties during training because the same input values to the inverse model will have different values at

Manuscript received July 21, 2007; revised November 13, 2007. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) and by COM DEV Ltd.

H. Kabir and Q.-J. Zhang are with the Department of Electronics, Carleton University, Ottawa, ON, Canada K1S 5B6 (e-mail: hkabir@doe.carleton.ca; qjz@doe.carleton.ca).

Y. Wang is with the Faculty of Engineering and Applied Sciences, University of Ontario Institute of Technology, Oshawa, ON, Canada L1H 7K4 (e-mail: Ying.Wang@uoit.ca).

M. Yu is with COM DEV Ltd., Cambridge, ON, Canada N1R 7H6 (e-mail: Ming.Yu@comdev.ca).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMTT.2008.919078

0018-9480/$25.00 © 2008 IEEE

the output (multivalued solutions). Consequently, the neural network inverse model cannot be trained accurately. This is why training an inverse model may become more challenging than training a forward model.

This paper considers the application of neural network inverse modeling techniques for microwave filter design. Some results have been reported using neural network techniques to model microwave filters including the rectangular waveguide iris bandpass filter [16]–[18], low-pass microstrip step filter [19], E-plane metal-insert filter [20], coupled microstrip line bandpass filter [21], etc. Waveguide dual-mode pseudoelliptic filters are often used in satellite applications due to its high Q, compact size, and sharp selectivity [22]. This particular filter holds complex characteristics whose conventional design procedure follows an iterative approach, which is time consuming. Moreover, the whole process has to be repeated even with a slight change in any of the design specifications. The modeling time increases as the filter order increases. Recently the neural network modeling technique has been applied to design a waveguide dual-mode pseudoelliptic filter [23]. By applying the neural network technique, filter design parameters were generated hundreds of times faster than EM-based models while retaining comparable accuracy.

The work in [23] primarily focused on how to apply the neural network to the waveguide dual mode pseudoelliptic filter design. However, it did not address the issues of inverse modeling, including nonuniqueness problems. The neural network inverse models in [23] were developed primarily using the direct method, although manual preprocessing of training data to take out observable contradictions partially dealt with the multivalued problem. The issues and problems in neural network inverse modeling still remain open and unsolved. This paper is a significant extension of the work presented in [23]. A new and systematic neural network inverse modeling methodology, completely beyond [23], is developed, and the problem of nonuniqueness in inverse modeling is formally addressed. The proposed methodology uses a set of novel criteria to detect multivalued solutions in training data, and uses adjoint neural network [8] derivative information to separate training data into groups, overcoming nonuniqueness problems in inverse models in a systematic way. Each group of data is used to train a separate inverse sub-model. Such inverse sub-models become more accurate since the individual groups of data do not have the problem of multivalued solutions. A complete methodology to solve the inverse modeling problem efficiently is proposed by combining various techniques including the direct inverse modeling, segmenting the inverse model, identifying multivalued solutions, dividing training data that have multivalued solutions, and combining separately trained inverse sub-models. A significant step is achieved beyond that of [23], where two actual filters are made following the neural network solutions, and real measurements from the filters are used to compare and validate the proposed neural network solutions.

Section II describes the formulation of the inverse models. The problem of nonuniqueness in the IO relationship is discussed. A method to check the existence of multivalued solutions in training data and a method to divide the data into groups are proposed. A method is proposed to combine the inverse sub-models to construct the overall inverse model. In Section III, a comprehensive methodology for neural network inverse modeling incorporating various steps is proposed. Practical examples of neural network inverse models for spiral inductor and various waveguide filter junctions are presented in Section IV. It also shows examples of cavity filter design using inverse models. The proposed approach is demonstrated to be faster than EM-based design and more accurate than the direct inverse modeling method.

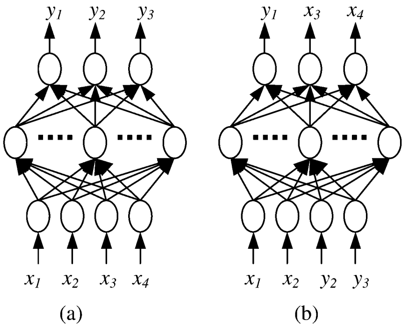


Fig. 1. Example illustrating neural network forward and inverse models. (a) Forward model. (b) Inverse model. The inputs x_3 and x_4 (output y_2 and y_3) of the forward model are swapped to the outputs (inputs) of the inverse model, respectively.

II. Inverse Modeling: Formulation and Proposed Neural Network Methods

A. Formulation

Let n and m represent the number of inputs and outputs of the forward model. Let $\boldsymbol{x}$ be an n-vector containing the inputs and $\boldsymbol{y}$ be an m-vector containing the outputs of the forward model. The forward modeling problem can then be expressed as

$$\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}) \tag{1}$$

where $\boldsymbol{x} = [x_1\ x_2\ x_3 \cdots x_n]^T, \boldsymbol{y} = [y_1\ y_2\ y_3 \cdots y_m]^T$, and $\boldsymbol{f}$ defines the IO relationship. An example of a neural network diagram of a forward model and its corresponding inverse model is shown in Fig. 1. Note that two outputs and two inputs of the forward model are swapped to the input and output of the inverse model, respectively. In general, some or all of them can be swapped from input to output or vice versa.

Let us define a subset of $\boldsymbol{x}$ and a subset of $\boldsymbol{y}$. These subsets of input and output are swapped to the output and input, respectively, in order to form the inverse model. Let I_x be defined as an index set containing the indices of inputs of the forward model that are moved to the output of the inverse model

$$I_x = \{i \mid \text{if } x_i \text{ becomes output of inverse model}\}. \tag{2}$$

Let I_y be the index set containing the indices of outputs of the forward model that are moved to the input of the inverse model

$$I_y = \{i \mid \text{if } y_i \text{ becomes input of inverse model}\}. \tag{3}$$

Let $\bar{\boldsymbol{x}}$ and $\bar{\boldsymbol{y}}$ be vectors of inputs and outputs of the inverse model. The inverse model can be defined as

$$\bar{\boldsymbol{y}} = \bar{\boldsymbol{f}}(\bar{\boldsymbol{x}}) \tag{4}$$

where $\bar{\boldsymbol{y}}$ includes y_i if $i \notin I_y$ and x_i if $i \in I_x$; $\bar{\boldsymbol{x}}$ includes x_i if $i \notin I_x$ and y_i if $i \in I_y$; and $\bar{\boldsymbol{f}}$ defines the IO relationship of the inverse model. For example, the inputs x_3 and x_4 of Fig. 1(a) may represent the iris length and width of a filter, and outputs y_2 and y_3 may represent the electrical parameter such as the coupling parameter and insertion phase. To formulate the inverse filter model, we swap the iris length and width with coupling parameter and insertion phase. For the example, in Fig. 1, the inverse model is formulated as

$$I_x = \{3, 4\} \tag{5}$$

$$I_y = \{2, 3\} \tag{6}$$

$$\begin{aligned}\bar{\boldsymbol{x}} &= [\bar{x}_1\ \bar{x}_2\ \bar{x}_3\ \bar{x}_4]^T \\ &= [x_1\ x_2\ y_2\ y_3]^T \end{aligned} \tag{7}$$

$$\begin{aligned}\bar{\boldsymbol{y}} &= [\bar{y}_1\ \bar{y}_2\ \bar{y}_3]^T \\ &= [y_1\ x_3\ x_4]^T. \end{aligned} \tag{8}$$

After formulation is finished, the model can be trained with the data. Usually data are generated by EM solvers originally in a forward way, i.e., given the iris length and compute coupling parameter. To train a neural network as an inverse model, we swap the generated data so that coupling parameter becomes training data for neural network inputs and iris length becomes training data for neural network outputs. The neural network trained this way is the direct inverse model.

The direct inverse modeling method is simple, and is suitable when the problem is relatively easy, e.g., when the original IO relationship is smooth and monotonic, and/or if the numbers of inputs/outputs are small. On the other hand, if the problem is complicated and the models using the direct method are not accurate enough, then segmentation of training data can be utilized to improve the model accuracy. Segmentation of microwave structures has been reported in existing literature such as [17] where a large device is segmented into smaller units. We apply the segmentation concepts over the range of model inputs to split the training data into multiple sections, each covering a smaller range of input parameter space. Neural network models are trained for each section of data. A small amount of overlapping data can be reserved between adjacent sections so that the connections between neighboring segmented models become smooth.

B. Nonuniqueness of IO Relationship in Inverse Model and Proposed Solutions

When the original forward IO relationship is not monotonic, the nonuniqueness becomes an inherent problem in the inverse model. In order to solve this problem, we start by addressing multivalued solutions in training data as follows. If two different input values in the forward model lead to the same value of output, then a contradiction arises in the training data of the inverse model because the single input value in the inverse model has two different output values. Since we cannot train the neural network inverse model to match two different output values simultaneously, the training error cannot be reduced to a small value. As a result, the trained inverse model will not be accurate. For this reason, it is important to detect the existence of multivalued solutions, which creates contradictions in training data.

Detection of multivalued solutions would have been straightforward if the training data were generated by deliberately choosing different geometrical dimensions such that they lead to the same electrical value. However, in practice, the training data are not sampled at exactly those locations. Therefore, we need to develop numerical criteria to detect the existence of multivalued solutions.

We assume I_x and I_y contain same amount of indices, and that the indices in I_x (or I_y) are in ascending order. Let us define the distance between two samples of training data, sample number l and k, as

$$d^{(k,l)} = \sqrt{\sum_{i=1}^{n} \left(\bar{x}_i^{(k)} - \bar{x}_i^{(l)}\right)^2 / \left(\bar{x}_i^{\max} - \bar{x}_i^{\min}\right)^2} \tag{9}$$

where $\bar{x}_i^{\max}$ and $\bar{x}_i^{\min}$ are the maximum and minimum value of $\bar{x}_i$, respectively, as determined from training data. We use a superscript to denote the sample index in training data. For example, $\bar{x}_i^{(k)}$ and $\bar{y}_i^{(k)}$ represent values of $\bar{x}_i$ and $\bar{y}_i$ in the kth training data, respectively. Sample $\bar{\boldsymbol{x}}^{(k)}$ is in the neighborhood of $\bar{\boldsymbol{x}}^{(l)}$ if $d^{(k,l)} < \varepsilon$, where ε is a user-defined threshold whose value depends on the step size of data sampling. The maximum and minimum "slope" between samples within the neighborhood of $\bar{\boldsymbol{x}}^{(l)}$ is defined as

$$\boldsymbol{S}_{\max}^{(l)} = \sqrt{\max_{\substack{k \\ d^{(k,l)} < \varepsilon}} \frac{\sum_{i=1}^{m} \left\{\left(\bar{y}_i^{(k)} - \bar{y}_i^{(l)}\right) / \left(\bar{y}_i^{\max} - \bar{y}_i^{\min}\right)\right\}^2}{\sum_{i=1}^{n} \left\{\left(\bar{x}_i^{(k)} - \bar{x}_i^{(l)}\right) / \left(\bar{x}_i^{\max} - \bar{x}_i^{\min}\right)\right\}^2}} \tag{10}$$

and

$$\boldsymbol{S}_{\min}^{(l)} = \sqrt{\min_{\substack{k \\ d^{(k,l)} < \varepsilon}} \frac{\sum_{i=1}^{m} \left\{\left(\bar{y}_i^{(k)} - \bar{y}_i^{(l)}\right) / \left(\bar{y}_i^{\max} - \bar{y}_i^{\min}\right)\right\}^2}{\sum_{i=1}^{n} \left\{\left(\bar{x}_i^{(k)} - \bar{x}_i^{(l)}\right) / \left(\bar{x}_i^{\max} - \bar{x}_i^{\min}\right)\right\}^2}}.$$

Input sample $\bar{\boldsymbol{x}}^{(l)}$ will have multivalued solutions if, within its neighborhood, the slope is larger than maximum allowed or the ratio of maximum and minimum slope is larger than the maximum allowed slope change. Mathematically, if

$$\boldsymbol{S}_{\max}^{(l)} > \boldsymbol{S}_M \tag{12}$$

and

$$\boldsymbol{S}_{\max}^{(l)} / S_{\min}^{(l)} > \boldsymbol{S}_R \tag{13}$$

then $\bar{\boldsymbol{x}}^{(l)}$ has multivalued solutions in its neighborhood where S_M is the maximum allowed slope and S_R is the maximum allowed slope change.

We employ the simple criteria of (12) and (13) to detect possible multivalued solutions. A suggestion for ε can be at least twice the average step size of y in training data. A reference value for S_R can be approximately the inverse of a similarly defined "slope" between adjacent samples in the training data of the forward model. The value of S_M should be greater than 1. In the overall modeling method, conservative choices of ε, S_M, and S_R (larger ε and smaller S_M and S_R) lead to more use of the derivative division procedure to be described in Section II-C, while aggressive choices of ε, S_M and S_R lead to early termination of the overall algorithm (or more use of the segmentation procedure) when model accuracy is achieved (or not achieved). In this way, the choices of ε, S_M and S_R mainly affect the training time of the inverse models rather than model accuracy. The modeling accuracy is determined from segmentation or from the derivative division step to be described in Section II-C. Sample values of ε, S_M, and S_R are given through an example in Section IV-C.

C. Proposed Method to Divide Training Data Containing Multivalued Solutions

If existence of multivalued solutions is detected in training data, we perform data preprocessing to divide the data into different groups such that the data in each group do not have the problem of multivalued solutions. To do this, we need to develop a method to decide which data samples should be moved into which group. We propose to divide the overall training data into groups based on derivatives of outputs versus inputs of the forward model. Let us define the derivatives of inputs and outputs that have been exchanged to formulate the inverse model, evaluated at each sample, as

$$\frac{\partial y_i}{\partial x_j}\Big|_{\boldsymbol{x}=\boldsymbol{x}^{(k)}}, \qquad i \in I_y \text{ and } j \in I_x \tag{14}$$

where $k = 1, 2, 3, \ldots, N_s$, and N_s is the total number of training samples. The entire training data should be divided based on the derivative criteria such that training samples satisfying

$$\frac{\partial y_i}{\partial x_j}\Big|_{\boldsymbol{x}=\boldsymbol{x}^{(k)}} < \boldsymbol{\delta} \tag{15}$$

belong to one group and training samples satisfying

$$\frac{\partial y_i}{\partial x_j}\Big|_{\boldsymbol{x}=\boldsymbol{x}^{(k)}} > -\boldsymbol{\delta} \tag{16}$$

belong to a different group. The value for δ is zero by default. However, to produce an overlapping connection at the break point between the two groups, we can choose a small positive value for it. In that case, a small amount of data samples whose absolute values of derivative are less than δ will belong to both groups. The value for δ other than the default suggestion of zero can be chosen as a value slightly larger than the smallest absolute value of derivatives of (14) for all training samples. Choice of δ only affects the accuracy of the sub-models at the connection region. The model accuracy for the rest of the region will remain unaffected.

This method exploits derivative information to divide the training data into groups. Therefore, accurate derivative is an important requirement for this method. Computation of derivatives of (14) is not a straightforward task since no analytical equation is available. We propose to compute the derivatives by exploiting the adjoint neural network technique [8]. We first train an accurate neural network forward model. After training is finished, its adjoint neural network can be used to produce the derivative information used in (15) and (16). The computed derivatives are employed to divide the training data into multiple smaller groups according to (15) and (16) using different combinations of i and j. Multiple neural networks are then trained with the divided data. Each neural network represents a sub-model of the overall inverse model.

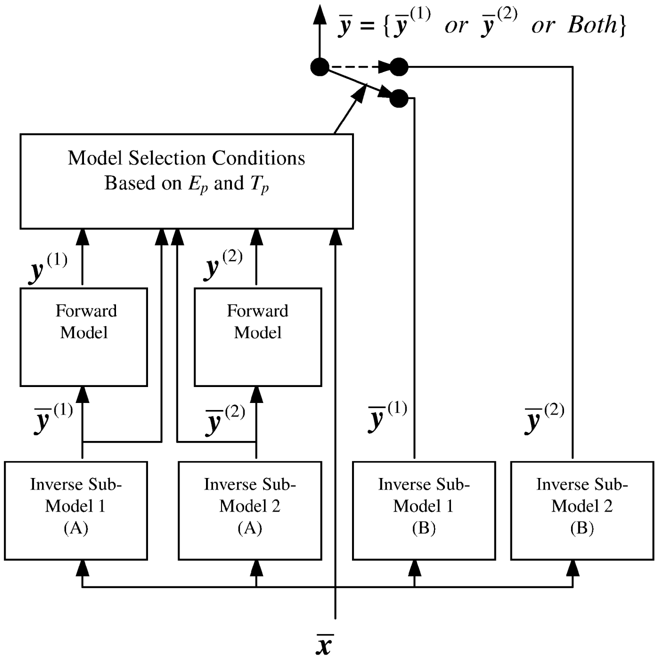


Fig. 2. Diagram of inverse sub-model combining technique after derivative division for a two sub-model system. Inverse sub-model 1 and inverse sub-model 2 in set (A) are competitively trained version of the inverse sub-models. Inverse sub-model 1 and inverse sub-model 2 in set (B) are trained with the divided data based on derivative criteria (15) and (16). The input and output of the overall combined model is $\bar{\boldsymbol{x}}$ and $\bar{\boldsymbol{y}}$, respectively.

Equations (12) and (13) play different roles versus (15) and (16) in our overall algorithm to be described in Section III. Equations (12) and (13) are used as simple and quick ways to detect the existence of contradictions in training data, but they do not give enough information on how the data should be divided. Equations (15) and (16), which require more computation (i.e., require training forward neural model) and produce more information, are used to perform detailed task of dividing training data into different groups to solve the multivalued problem.

D. Proposed Method to Combine the Inverse Sub-Models

We need to combine the multiple inverse sub-models to reproduce the overall inverse model completely. For this purpose, a mechanism is needed to select the right one among multiple inverse sub-models for a given input $\bar{\boldsymbol{x}}$. Fig. 2 shows the proposed inverse sub-model combining method for a two sub-model system. For convenience of explanation, suppose $\bar{\boldsymbol{x}}$ is a randomly selected sample of training data. Ideally if $\bar{\boldsymbol{x}}$ belongs to a particular inverse sub-model, then the output from it should be the most accurate one among various inverse sub-models. Conversely, the outputs from the other inverse sub-models should be less accurate if $\bar{\boldsymbol{x}}$ does not belong to them. However, when using the inverse sub-models with general input $\bar{\boldsymbol{x}}$ whose values are not necessarily equal to that of any training samples, the value from the sub-models is the unknown parameter to be solved. Thus, we still do not know which inverse sub-model is the most accurate one.

To address this dilemma, we use the forward model to help deciding which inverse sub-model should be selected. If we supply an output from the correct inverse sub-model to an accurate forward model, we should be able to obtain the original data input to the inverse sub-model. For example, suppose $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$ is an accurate forward model. Suppose the inputs and outputs of the inverse sub-model are defined such that $\bar{\boldsymbol{x}} = \boldsymbol{y}$ and $\bar{\boldsymbol{y}} = \boldsymbol{x}$. If the inverse sub-model $\bar{\boldsymbol{y}} = \bar{\boldsymbol{f}}(\bar{\boldsymbol{x}})$ is true, then

$$\boldsymbol{f}(\bar{\boldsymbol{f}}(\bar{\boldsymbol{x}})) = \bar{\boldsymbol{x}} \tag{17}$$

is also true. Conversely, if $\boldsymbol{f}(\bar{\boldsymbol{f}}(\bar{\boldsymbol{x}})) \neq \bar{\boldsymbol{x}}$ then, $\bar{\boldsymbol{f}}(\bar{\boldsymbol{x}})$ is a wrong inverse sub-model. In this way, we can use a forward model to help determining which inverse sub-model should be selected for a particular value of input. In our method, input $\bar{\boldsymbol{x}}$ is supplied to each inverse sub-model and output from them is fed to the accurately trained forward model, respectively, which generates different $\boldsymbol{y}$. These outputs are then compared with the input data $\bar{\boldsymbol{x}}$. The inverse sub-model that produces least error between $\boldsymbol{y}$ and $\bar{\boldsymbol{x}}$ is selected and the output from the corresponding inverse sub-model is chosen as the final output of the overall inverse modeling problem.

Let us assume an inverse model is divided into N different inverse sub-models according to derivative criteria. The error between the input of the pth inverse sub-model and output of the forward model (also called error from the inverse-forward sub-model pair) is calculated as

$$E_p = \frac{1}{2}\sqrt{\sum_{\substack{i\in I_y\\ j\in I_x}} \left(y_i^{(p)} - \bar{x}_j\right)^2 / \left(\bar{x}_j^{\max} - \bar{x}_j^{\min}\right)^2} \tag{18}$$

where $p = 1, 2, 3, \ldots, N$, and we have assumed I_y and I_x contain equal amount of indices. N is the number of sub models. As an example, E_1 would be lower than $E_2, E_3, \ldots, E_N$ if a sample $\bar{\boldsymbol{x}}$ belongs to the inverse sub-model 1.

We include another constraint to the inverse sub-model selection criteria. This constraint checks for the training range. If an inverse sub-model produces an output that is located outside its training range, then the corresponding output is not selected even though the error (E_p) of (18) is less than that of other inverse sub-models. If the outputs of other inverse sub-models are also found outside their training range, then we compare their magnitude of distances from the boundary of training range. An inverse sub-model producing the shortest distance is selected in this case. For sub-model p, the distance of a particular output outside the training range can be defined as

$$\boldsymbol{T}_i^{(p)} = \begin{cases} \bar{y}_i^{(p)} - \bar{y}_i^{\max}, & \text{for } \bar{y}_i^{(p)} > \bar{y}_i^{\max} \\ \bar{y}_i^{\min} - \bar{y}_i^{(p)}, & \text{for } \bar{y}_i^{(p)} < \bar{y}_i^{\min} \\ 0, & \text{otherwise} \end{cases} \tag{19}$$

where $i \in I_x, p = 1, 2, 3, \ldots, N$, and $\bar{y}_i^{\max}$ and $\bar{y}_i^{\min}$ are the maximum and minimum values of $\bar{y}_i$, respectively, obtained from the training data. For any output $\bar{\boldsymbol{y}}$, if the distance is zero, then the output is located inside the training range. The total distance outside the range for all the outputs of an inverse sub-model p can be calculated as

$$\boldsymbol{T}_p = \sum_{i\in I_x} \boldsymbol{T}_i^{(p)} \tag{20}$$

where $i \in I_x$ and $p = 1, 2, 3, \ldots, N$.

The calculated E_p and T_p are used to determine which inverse sub-model should be selected for a particular set of input. The inverse sub-model selection criteria can be expressed as

$$\bar{y} = \bar{y}^{(p)} \tag{21}$$

if $(T_p = 0)$ *AND* $(T_q = 0)$ *AND* $(E_p < E_q)$, or $((T_p \neq 0)$ *OR* $(T_q \neq 0))$ *AND* $(T_p < T_q)$ for all values of q, where $q = 1, 2, 3, \ldots, N$ and $q \neq p$. For example, inverse sub-model 1 is selected if outputs from all the inverse sub-models are located inside the training range and the error produced by the inverse-forward sub-pair 1 is less than the error produced by all other pairs, or if the output of any of the inverse sub-model is located outside the training range and the distance of the output of inverse sub-model 1 is the least of that of all other inverse sub-models.

In cases when outputs from multiple inverse sub-models remain inside the training range (i.e., $T_p = 0$) and at the same time the errors (i.e., E_p) calculated from the corresponding inverse-forward pairs are all smaller than a threshold value (E_T), then the outputs of those inverse sub-models are valid solutions. As an example, suppose we have three inverse sub-models $(N = 3)$. For a particular sample of data, if the outputs from inverse sub-model 1 and inverse sub-model 2 both fall within the training range $(T_1 = T_2 = 0)$ and the errors E_1 and E_2 are both less than the threshold error E_T, then solutions from inverse sub-model 1 and inverse sub-model 2 are both accepted.

The purpose of the model-combining technique is to reproduce the original multivalued IO relationship for the user. Our method is an improvement over the direct inverse modeling method since the latter produces only an inaccurate result in case there are multivalued solutions (i.e., produces a single solution, which may not match any of the original multivalues). Our method can be used to provide a quick model to reproduce multivalued solutions in inverse EM problems. Using the solutions from the proposed inverse model (including reproduced multivalued solutions), the user can proceed to circuit design.

E. Accuracy Enhancement of Sub-Model Combining Method

Here we describe two ways to further enhance the selection and, thus, improve the accuracy of the overall inverse model. These enhancement techniques are used only for some subregions where model selections are inaccurate. In most cases, regularly trained inverse sub-models will be accurate with no need of these enhancement techniques. The subregions that need enhancement can be determined by checking the model selection using the known divisions in training data. The application of the enhancement techniques will incrementally increase model development time.

1) Competitively Trained Inverse Sub-Model: To further improve the inverse sub-model selection accuracy, an additional set of competitively trained inverse sub-models can be used. These inverse sub-models are trained to learn not only what is correct, but also what is wrong. Correct data are the data that belong only to a particular inverse sub-model. Conversely, incorrect data are the data in which $\bar{\boldsymbol{x}}$ belongs to other inverse sub-models and $\bar{\boldsymbol{y}}$ is deliberately set to zero so that the inverse sub-model is forced to learn wrong values of $\bar{\boldsymbol{y}}$ for $\bar{\boldsymbol{x}}$ that do not belong to this inverse sub-model. The output values of these inverse sub-models are not very accurate, but they are reliable to identify if an input either belongs or does not belong to the inverse sub-model. Therefore, they are used for the inverse sub-model selection purpose only. Once the selection has been made, the final output is taken from the regularly trained (i.e., not competitively trained) inverse sub-model. In Fig. 2, the inverse sub-models in set (A) represent the competitively trained inverse sub-models and set (B) represents regularly trained inverse sub-models.

2) Forward Sub-Model: The default forward model used in the model combining method is trained with the entire set of training data. The decision of choosing the right inverse sub-model depends on the accuracy of both inverse sub-models and forward models. We can further tighten the accuracy of the forward model by training multiple forward sub-models using the same groups of data used to train inverse sub-models. These forward sub-models capture the same data range as its inverse counterpart and, therefore, the inverse and forward sub-model pairs are capable of producing more accurate decision. In Fig. 2, the forward models are replaced with the forward sub-models.

III. Overall Inverse Modeling Methodology

The overall methodology of inverse modeling combines all the aspects described in Section II. The inverse model of a microwave device may contain unique or nonunique behavior over various regions of interest. In the region with unique solutions, direct segmentation can be applied and training error is expected to be low. On the other hand, in the region with nonuniqueness, the model should be divided according to derivative. If the overall problem is simple, the methodology will end with a simple inverse model directly trained with all data. In complicated cases, the methodology uses a derivative division and sub-model combining method to increase model accuracy. This approach increases the overall efficiency of modeling. The flow diagram of the overall inverse modeling approach is presented in Fig. 3. The overall methodology is summarized in the following steps.

Step 1) Define the inputs and outputs of the model. Detailed formulation can be found in Section II-A. Generate data using EM simulator or measurement. Swap the input and output data to obtain data for training inverse model. Train and test the inverse model. If the model accuracy is satisfied, then stop. Results obtained here is the direct inverse model.

Step 2) Segment the training data into smaller sections. If there have been several consecutive iterations between Steps 2) and 5), then go to Step 6).

Step 3) Train and test models individually with segmented data.

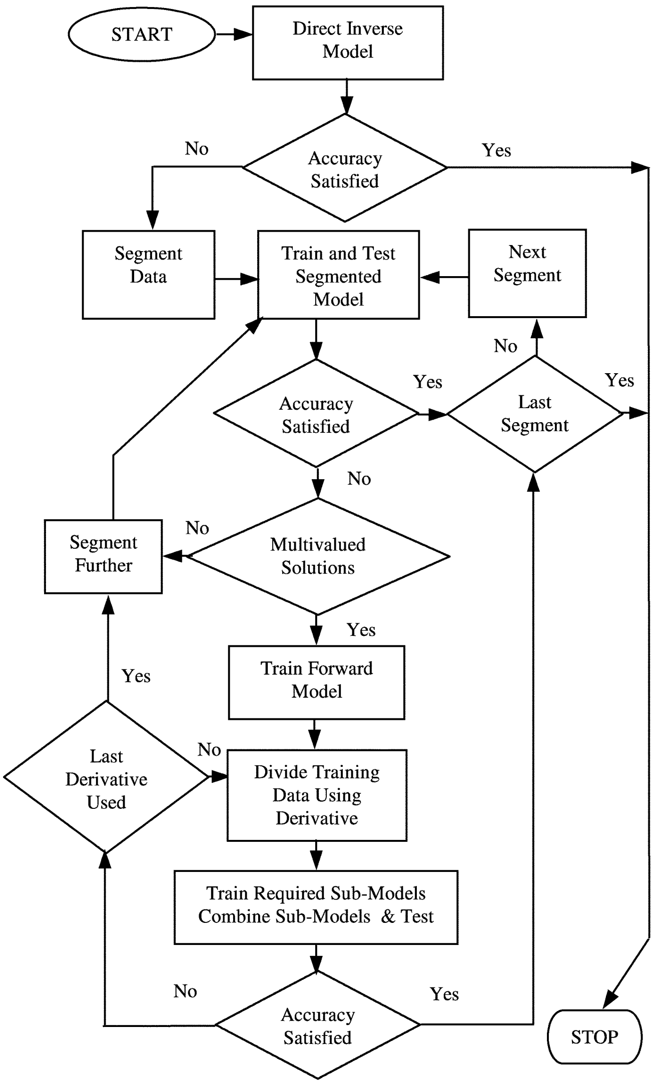


Fig. 3. Flow diagram of overall inverse modeling methodology consisting of direct, segmentation, derivative dividing, and model combining techniques.

Step 4) If the accuracy of all the segmented models in Step 3) is satisfied, stop. Else for the segments that have not reached accuracy requirements, proceed to the next steps.

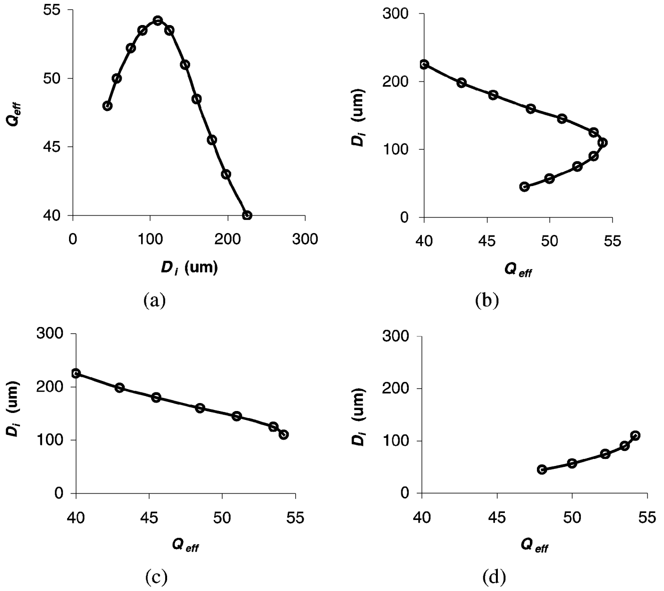
Step 5) Check for multivalued solutions in model's training data using (12) and (13). If none are found, then perform further segmentation by going to Step 2).

Step 6) Train a neural network forward model.

Step 7) Using the adjoint neural network of the forward model, divide the training data according to derivative criteria, as described in Section II-C.

Step 8) With the divided data, train necessary sub-models, for example, two inverse sub-models. Optionally obtain two competitively trained inverse sub-models and two forward sub-models.

Step 9) Combine all the sub-models that have been trained in Step 8) according to the method in Section II-D. Test the combined inverse sub-models. If the test accuracy is achieved, then stop. Else go to Step 7) for



(a) (b) (c) (d)

Fig. 4. Nonuniqueness of IO relationship is observed when $Q_{\rm eff}$ versus D_i data of a forward spiral inductor model is exchanged to formulate an inverse model. (a) Unique relationship between input and output of a forward model. (b) Nonunique relationship of IO of an inverse model obtained from forward model of (a). Training data containing multivalued solutions of Fig. 4(b) are divided into groups according to derivative. (c) Group I data with negative derivative. (d) Group II data with positive derivative. Within each group, the data are free of multivalued solutions, and consequently, the IO relationship becomes unique.

further division of data according to derivative information in different dimensions, or if all the dimensions are exhausted, go to Step 2).

The algorithm increases efficiency by choosing the right techniques in the right order. For simple problems, the algorithm stops immediately after the direct inverse modeling technique. In this case, no data segmentation or other techniques are used, and training time is short. The segmentation and subsequent techniques will be applied only when the directly trained model cannot meet the accuracy criteria. In this way, more training time is needed only with more complexity in the model IO relationship, such as the multivalued relationship.

IV. Examples and Applications to Filter Design

A. Example 1: Inverse Spiral Inductor Model

In this example, we illustrate the proposed technique through a spiral inductor modeling problem where the input of the forward model is the inner mean diameter (D_i) of the inductor, and the output is the effective quality factor $(Q_{\rm eff})$. Fig. 4(a) shows the variation of $Q_{\rm eff}$ with respect to inner diameter [24]. The inverse model of this problem is shown in Fig. 4(b), which shows a nonunique IO relationship since, in the range from $Q_{\rm eff} = 47$ to $Q_{\rm eff} = 55$, a single $Q_{\rm eff}$ value will produce two different D_i values. We have implemented (10)–(13) in NeuroModelerPlus [25] to detect the existence of multivalued solutions, as described in Section II-B. We supply the training data to NeuroModelerPlus and set values of parameters as $S_M = 80, S_R = 80$, and $\varepsilon = 0.01$. The program detects several contradictions in the data.

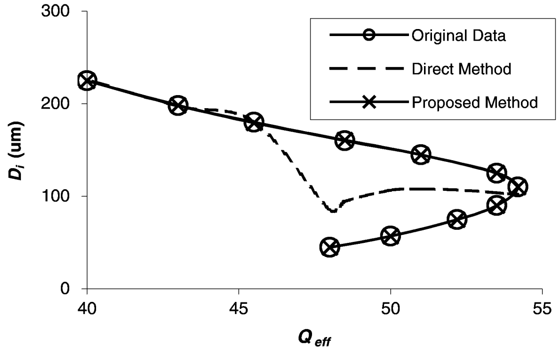


Fig. 5. Comparison of inverse model using the proposed methodology and the direct inverse modeling method for the spiral inductor example.

In the next step, we divide the training data according to the derivative. We trained a neural network forward model to learn the data in Fig. 4(a) and used its adjoint neural network to compute the derivatives $(\partial Q_{\rm eff}/\partial D_i)$. We compared all the values of derivatives and the lowest absolute value was found to be 0.018. The next large absolute value of the derivative was 0.07. Therefore, we chose the value of $\delta = 0.02$, which is in from 0.018 to 0.07. The training data are divided such that samples satisfying (15) are divided into group I and samples satisfying (16) are divided into group II. Fig. 4(c) and (d) shows the plots of two divided groups, which confirm that the individual groups become free of multivalued solutions after dividing the data according to the derivative information.

Two inverse sub-models of the spiral inductor were trained using the divided data of Fig. 4(c) and (d). The two individual sub-models became very accurate and they were combined using the model combining technique. For comparative purposes, a separate model was trained using the direct inverse modeling method, which means that all the training samples in Fig. 4(b) were used without any data division to train a single inverse model. The results are shown in Fig. 5. It shows that the model obtained from the direct inverse modeling method produce an inaccurate result because of confusions over training data with multivalued solutions. The model trained using the proposed methodology delivers accurate solutions that match the data for the entire range. Average test error reduced from 13.6% down to 0.05% using proposed techniques over the direct inverse modeling method.

B. Example 2: Filter Design Approach and Development of Inverse Internal Coupling Iris and IO Iris Models

Neural network modeling techniques are applied to the microwave waveguide filter design. The filter design starts from synthesizing the coupling matrix to satisfy ideal filter specifications. The EM method for finding physical/geometrical parameters to realize the required coupling matrix is an iterative EM optimization procedure. In our examples, this procedure performs EM analysis (mode-matching or finite-element methods) on each waveguide junction of the filter to get the generalized scattering matrix (GSM). From the GSM, we extract coupling coefficients. We then modify the design parameters (i.e., the dimensions of the filter) and re-perform EM analysis iteratively

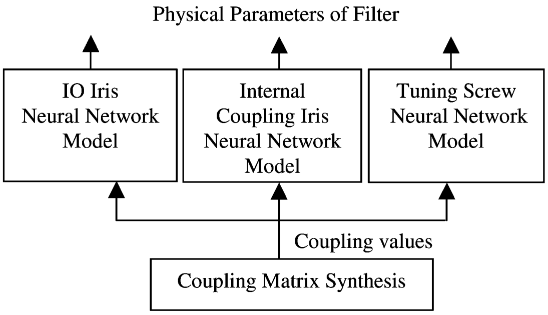

Physical Parameters of Filter
IO Iris Neural Network Model
Internal Coupling Iris Neural Network Model
Tuning Screw Neural Network Model
Coupling values
Coupling Matrix Synthesis

Fig. 6. Diagram of the filter design approach using the neural network inverse models.

until the required coupling coefficients are realized. In our proposed approach, we avoid this iterative step and use neural network inverse models to directly provide the filter dimensions.

In this study, the filter was decomposed into three different modules, each representing a separate filter junction. Neural network inverse models of these junctions were developed separately using the proposed methodology. The three models are the IO iris, the internal coupling iris, and the tuning screws. Training data for neural networks are generated from physical parameters firstly through EM simulation (mode-matching method) producing the GSM. Coupling values are then obtained from the GSM through analytical equations. Fig. 6 demonstrates the filter design approach. More detailed information on modeling and the design procedure for the filter can be found in [23].

In this example, we develop two inverse neural network models for the waveguide filter. The first neural network inverse model of the filter structure is developed for the internal coupling iris. The inputs and outputs of the internal coupling iris forward model are

$$\boldsymbol{x} = [D \quad fo \quad L_v \quad L_h]^T \tag{22}$$

$$\boldsymbol{y} = [M_{23} \quad M_{14} \quad P_v \quad P_h]^T \tag{23}$$

where D is the circular cavity diameter, fo is the center frequency, M_{23} and M_{14} are coupling values, L_v and L_h are the vertical and horizontal coupling slot lengths, and P_v and P_h are the loading effect of the coupling iris on the two orthogonal modes, respectively. The inverse model is formulated as

$$\bar{\boldsymbol{y}} = [x_3 \quad x_4 \quad y_3 \quad y_4]^T = [L_v \quad L_h \quad P_v \quad P_h]^T \tag{24}$$

$$\bar{\boldsymbol{x}} = [x_1 \quad x_2 \quad y_1 \quad y_2]^T = [D \quad fo \quad M_{23} \quad M_{14}]^T. \tag{25}$$

The second inverse model of the filter is the IO iris model. The input parameters of the IO iris inverse model are circular cavity diameter D, center frequency fo, and the coupling value R. The output parameters of the model are the iris length L, the loading effect of the coupling iris on the two orthogonal modes P_v and P_h, and the phase loading on the input rectangular waveguide P_{in}. The IO iris forward model is formulated as

$$\boldsymbol{x} = [D \quad fo \quad L]^T \tag{26}$$

$$\boldsymbol{y} = [R \quad P_v \quad P_h \quad P_{\text{in}}]^T. \tag{27}$$

The inverse model is defined as

$$\bar{\boldsymbol{y}} = [x_3 \quad y_2 \quad y_3 \quad y_4]^T = [L \quad P_v \quad P_h \quad P_{\text{in}}]^T \tag{28}$$

$$\bar{\boldsymbol{x}} = [x_1 \quad x_2 \quad y_1]^T = [D \quad fo \quad R]^T. \tag{29}$$

Training data were generated in the forward way (according to forward model) and the data are then reorganized for training inverse model. The entire data was used to train the inverse internal coupling iris model. For the IO iris model, four different sets of training data were generated according to the width of iris using the mode-matching method. The model for each set was trained and tested separately using the direct inverse modeling method. For both of the iris models, direct training produced good accuracy in terms of average and $L2$ (least squares [2]) errors. However the worst case errors were large. Therefore, in the next step, the data was segmented into smaller sections. Models for these sections were trained separately, which reduced the worst case error. The final model results of the coupling iris model shows that the average error reduced from 0.24% to 0.17% and worst case error reduced from 14.2% to 7.2%. The average error for the IO iris model reduced from 1.2% to 0.4% and worst case error reduced from 54% to 18.4%. The errors for other sets of the IO iris model also reduced similarly. We can improve the accuracy further by splitting the data set into more sections and achieve accurate results as required. In this example, our methodology stops with accurate inverse model at Step 4) without derivative division of data. These models are developed using proposed methodology and provide better accuracy than models developed using the direct method.

C. *Example 3: Inverse Tuning Screw Model*

The last neural network inverse model of the filter is developed for the tuning screw model. This model has complicated IO relationships requiring the full algorithm to be applied. Here we describe this example in detail. The model outputs are the phase shift of the horizontal mode across the tuning screw P_h, coupling screw length L_c, and the horizontal tuning screw length L_h. The input parameters of this model are circular cavity diameter D, center frequency fo, the coupling between the two orthogonal modes in one cavity M_{12}, and the difference between the phase shift of the vertical mode and that of the horizontal mode across the tuning screw P. The forward tuning screw model is defined as

$$\boldsymbol{x} = [D \quad fo \quad L_h \quad L_C]^T \tag{30}$$

$$\boldsymbol{y} = [M_{12} \quad P \quad P_h]^T. \tag{31}$$

The inverse model is formulated as

$$\bar{\boldsymbol{y}} = [y_3 \quad x_3 \quad x_4]^T = [P_h \quad L_h \quad L_C]^T \tag{32}$$

$$\bar{\boldsymbol{x}} = [x_1 \quad x_2 \quad y_1 \quad y_2]^T = [D \quad fo \quad M_{12} \quad P]^T. \tag{33}$$

In the initial step, the inverse model was trained directly using entire training data. The training error was high even with many hidden neurons. Therefore, we proceed to segment the data into smaller sections. In this example, we used the segmentation, which corresponds to two adjacent samples of frequency fo and two adjacent samples of diameter D. Each segment of data was

TABLE I
COMPARISON OF MODEL TEST ERRORS BETWEEN DIRECT AND PROPOSED METHODS FOR TUNING SCREW MODEL

Inverse Modeling Method	Model Test Error (%)		
	Average	L2	Worst
Direct	3.85	7.51	94.25
Proposed	0.40	0.59	5.10

used to train a separate inverse model. Some of the segments produced accurate models with error less than 1%, while others were still inaccurate.

The segments that could not reach the desired accuracy were checked for the existence of multivalued solutions individually. The method to check the existence of multivalued solutions using (10)–(13), as described in Section II-B, has been implemented in NeuroModelerPlus [25] software. We use this program to detect the existence of multivalued solutions in the training data. For this example, neighborhood size $\varepsilon = 0.01$, maximum slope $S_M = 80$, and maximum slope change $S_R = 80$ were chosen. NeuroModelerPlus suggests that the data contain multivalued solutions. Therefore, we need to proceed to train a neural network forward model and apply the derivative division technique to divide the data.

To compute the derivative, we trained a neural network as a forward tuning screw model. Then derivatives were computed using an adjoint neural network model through NeuroModelerPlus. Considering $\delta = 0$ and applying the derivative $(\partial P/\partial L_h)$ to (15) and (16), we divided the data into groups I and II, respectively. Two inverse sub-models were trained using groups I and II data. As in Step 8) of the methodology, we trained two forward sub-models using data of groups I and II. The equations for error criteria E_1 and E_2, distance criteria T_1 and T_2, and model selection can be obtained using (18), (20), and (21), respectively.

The entire process was done using NeuroModelerPlus. The segments that failed to reach good accuracy before became more than 99% accurate after a derivative division and model combining technique were applied. The process was continued until all data were captured. A few of the sub-models needed the accuracy enhancement techniques to select the right models and, thus, reach the desired accuracy. The result of the inverse model using the proposed methodology is compared with the direct inverse method in Table I, showing the average, $L2$, and worst case errors between the model and test data. Table I demonstrates that the proposed methodology produces significantly better results than the direct method.

Fig. 7 shows the plot of phase (P) for various horizontal screw lengths (L_h), which defines the forward model relationship. The two curves in the figure represent forward training data at two different frequencies. The forward relationship is unique, which means that there are no multivalued solutions. Fig. 8(a) and (b) shows the outputs of two inverse models trained using a direct and proposed methodology where the output and input are L_h and P, respectively, for two different frequencies. The data of the two plots represent the same data as that in Fig. 7, except the input and output are swapped. The inverse training data in both plots of Fig. 8(a) and (b) contain multivalued solutions, and it is clear from the two plots that the inverse model trained using the direct method cannot match the data, whereas the inverse model using the proposed methodology produce the output L_h very accurately for the entire range. To demonstrate the variation of multivalued problem at the different cavity diameter D, we show two more plots in Fig. 8(c) and (d). They correspond to two different diameters at the same frequency. Fig. 8(c) contains multivalued data, whereas Fig. 8(d) does not contain any multivalued data. The plots also compare the outputs of the proposed method and direct method. From Fig. 8(d), we can see that, for single valued case, both methods produce acceptable results, whereas in the multivalued case [see Fig. 8(c)], only the proposed model can produce accurate results. In reality,

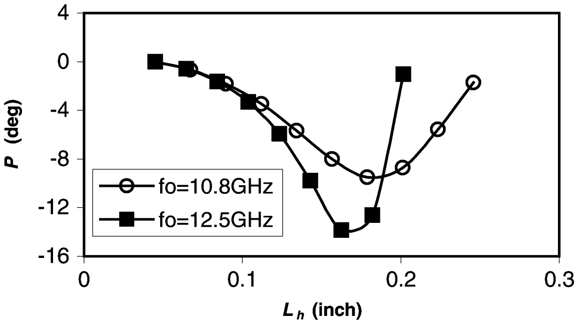


Fig. 7. Original data showing variation of phase angle (P) with respect to horizontal screw length (L_h) describing unique relationship of forward tuning screw model.

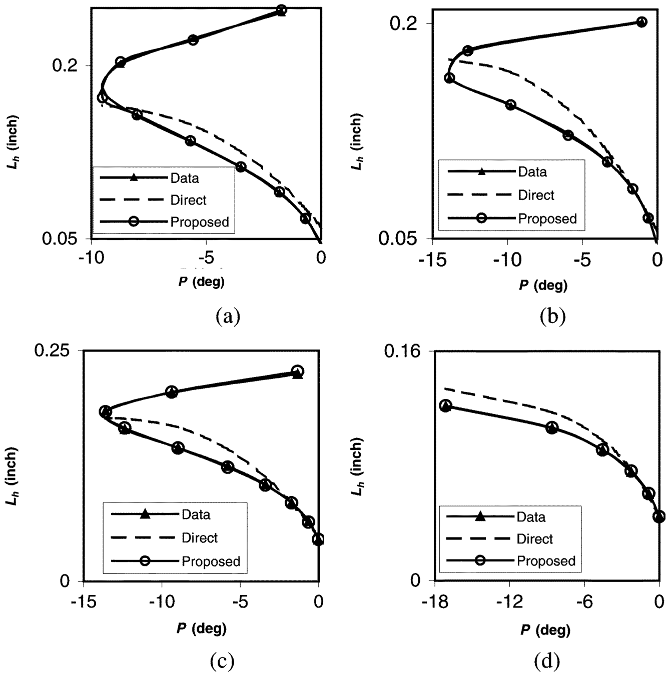


Fig. 8. Comparison of output (L_h) of inverse tuning screw model trained using direct and proposed methods at two different frequencies: (a) $fo = 10.8$ GHz, $D = 1.11$ in and (b) $fo = 12.5$ GHz, $D = 1.11$ in. It is evident that this inverse model has nonunique outputs. The proposed method produced a more accurate inverse model than that of the direct inverse method. Inverse data are plotted for two different diameters: (c) $fo = 11.85$ GHz, $D = 1.09$ and (d) $fo = 11.85$, $D = 0.95$. Fig. 8(c) contains multivalued data, whereas Fig. 8(d) does not contain any multivalued data. This demonstrates the necessity of automatic algorithms to detect and handle multivalued scenarios in different regions of the modeling problem.

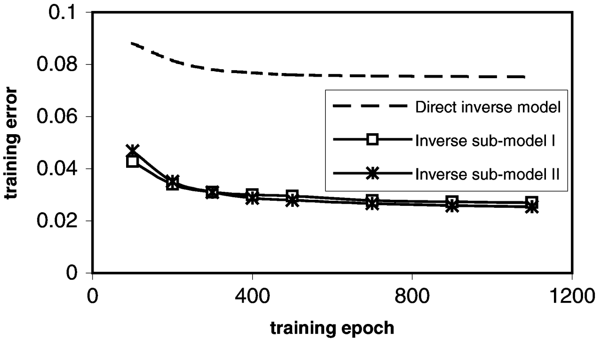


Fig. 9. Training error of inverse tuning screw model following direct inverse modeling approach and proposed derivative division approach. The training errors of both the inverse sub-models are lower than that of direct inverse model.

it is not known beforehand which region contains multivalued data and which region does not. This is why the proposed algorithm is useful to automatically detect the regions that contain multivalued data and apply the appropriate techniques in that region to improve accuracy. In this way, model development can be performed more systematically via computer.

As an additional demonstration of the usefulness of derivative division, we applied the same derivative as that described earlier in this section to the entire training data and divided the data into two groups (containing 28 000 and 6000 samples, respectively) according to (15) and (16). The training errors of the individual inverse sub-models are compared with that of the direct inverse model in Fig. 9, which shows that derivative division technique significantly reduces the training error. The test errors are similar as training error in this example. The training epoch in Fig. 9 is defined as one iteration of training when all training data have been used to make an update of neural network weights [1].

D. Example 4: Four-Pole Filter Design for Device Level Verification Using the Three Developed Inverse Models

In this example, we use the neural network inverse models that were developed in Examples 2 and 3 to design a four-pole filter with two transmission zeros. Compared to the example in [23], which shows the simulation results only, the present example describes new progress, where the filter results are used to fabricate an actual filter and real measurement data are used to validate the neural network solutions.

The layout of a four-pole filter is similar to that in [23]. The filter center frequency is 11.06 GHz, bandwidth is 58 MHz, and the cavity diameter is chosen to be 1.17 in. The normalized ideal coupling values are

$$R_1 = R_2 = 1.07$$

$$M = \begin{bmatrix} 0 & 0.86 & 0 & -0.278 \\ 0.86 & 0 & 0.82 & 0 \\ 0 & 0.82 & 0 & 0.86 \\ -0.278 & 0 & 0.86 & 0 \end{bmatrix}. \qquad (34)$$

The trained neural network inverse models developed in Examples 2 and 3 are used to calculate irises and tuning screw dimensions. The filter is manufactured and then tuned by adjusting irises and tuning screws to match the ideal response. Fig. 10 compares the measured and ideal filter responses. Dimensions are listed in Table II. A very good correlation can be seen between the initial dimensions provided by the neural network inverse models and the measured final dimensions of the fine tuned filter.

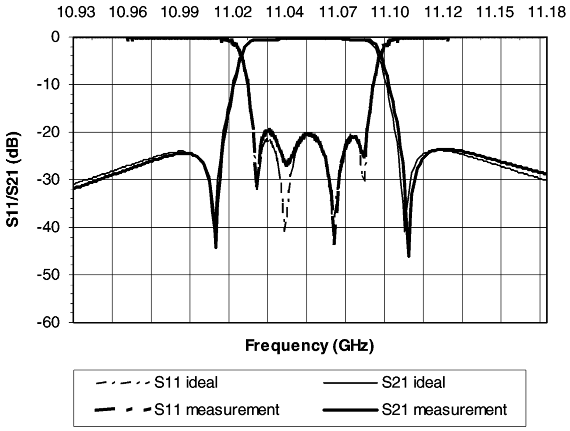


Fig. 10. Comparison of the ideal four-pole filter response with the measured filter response after tuning. The dimensions of the measured filter were obtained from neural network inverse models.

TABLE II
COMPARISON OF DIMENSIONS OF THE FOUR-POLE FILTER OBTAINED BY THE NEURAL NETWORK INVERSE MODEL AND MEASUREMENT

	Neural Model (inch)	Measurement (inch)	Difference (inch)
IO irises	0.405	0.405	0
M_{23} iris	0.299	0.297	-0.002
M_{14} iris	0.212	0.216	0.004
M_{11}/M_{44} tuning screws	0.045	0.005	-0.040
M_{22}/M_{33} tuning screws	0.133	0.135	0.002
M_{12}/M_{34} coupling screws	0.111	0.115	0.004
Cavity length	1.865	1.864	-0.001

E. Example 5: Six-Pole Filter Design for Device Level Verification of Proposed Methods

In this example, we design a six-pole waveguide filer using the proposed methodology. The specification of this six-pole filter is different from that of Example 4. The filter center frequency is 12.155 GHz, bandwidth is 64 MHz, and cavity diameter is chosen to be 1.072 in. This filter is higher in order and more complex in nature than that of Example 4. This filter uses an additional iris called a "slot iris." For this reason, in addition to the neural models of Examples 2 and 3, we developed another inverse model for slot iris. The inputs of the slot iris model are cavity diameter D, center frequency fo, and coupling M, and the outputs are iris length L, vertical phase P_v, and horizontal phase P_h. This model and the other three neural network inverse models developed in Examples 2 and 3 were used to design a

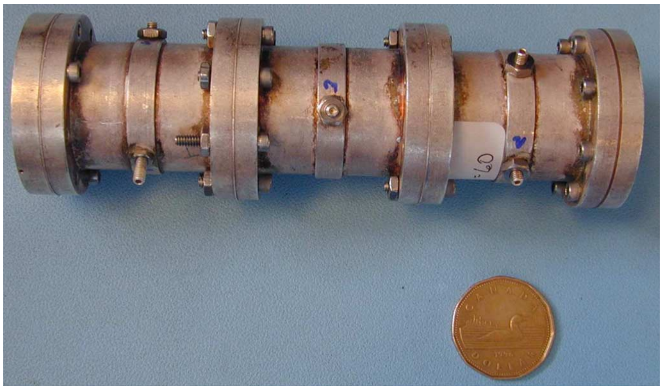


Fig. 11. Six-pole waveguide filter designed and fabricated using the proposed neural network method.

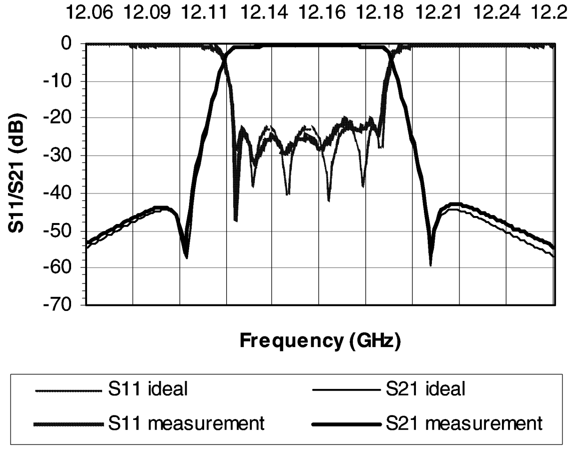


Fig. 12. Comparison of the six-pole filter response with ideal filter response. The filter was designed, fabricated, tuned, and then measured to obtain the dimensions.

filter. This filter is fabricated and measured for device-level verification. The normalized ideal coupling values are

$$R_1 = R_2 = 1.077$$

$$M = \begin{bmatrix} 0 & 0.855 & 0 & -0.16 & 0 & 0 \\ 0.855 & 0 & 0.719 & 0 & 0 & 0 \\ 0 & 0.719 & 0 & 0.558 & 0 & 0 \\ -0.16 & 0 & 0.558 & 0 & 0.614 & 0 \\ 0 & 0 & 0 & 0.614 & 0 & 0.87 \\ 0 & 0 & 0 & 0 & 0.87 & 0 \end{bmatrix}. \tag{35}$$

After obtaining the filter dimensions from the inverse neural network models, we manufactured the filter and tuned it by adjusting irises and tuning screws to match the ideal response. The picture of the fabricated filter is shown in Fig. 11. Fig. 12 presents the response of the tuned filter and compares it with the ideal one, showing a perfect match between each other. The dimensions of the tuned filter are measured and compared with the dimensions obtained from the neural network inverse models in Table III, along with EM design results. From Table III, we see that the neural network dimensions match the measurement dimensions very well. The quality of the solutions from the inverse neural networks is similar to that from the EM design, both being excellent starting points for final tuning of the filter. The biggest error of screw dimensions, common for both the inverse neural network solution and the EM design, is observed in cavity 2, which is caused by the manufacturing error. The cavity length was manufactured short by 0.003 in and that error affected the screw dimensions. In other words, this error was compensated by tuning.

TABLE III
COMPARISON OF DIMENSIONS OBTAINED BY THE EM MODEL, THE NEURAL NETWORK INVERSE MODEL, AND THE MEASUREMENT OF THE TUNED SIX-POLE FILTER

Filter Dimensions	EM model (inch)	Neural model (inch)	Measurement (inch)
IO irises	0.352	0.351	0.358
M23 iris	0.273	0.274	0.277
M14 iris	0.167	0.170	0.187
M45 iris	0.261	0.261	0.262
Cavity 1 length	1.690	1.691	1.690
Tuning screw	0.079	0.076	0.085
Coupling screw	0.097	0.097	0.104
Cavity 2 length	1.709	1.709	1.706
Tuning screw	0.055	0.045	0.109
Coupling screw	0.083	0.082	0.085
Cavity 3 length	1.692	1.692	1.692
Tuning screw	0.067	0.076	0.078
Coupling screw	0.098	0.097	0.120

TABLE IV
COMPARISON OF TIME TO OBTAIN THE DIMENSIONS BY NEURAL NETWORK INVERSE MODELS AND EM MODELS

Method	Time (s)		
	IO iris	Coupling iris	Tuning screw
EM	15	120	240
Neural network	0.14E-3	0.1E-3	1.3E-3

The advantage of using the trained neural network inverse models is also realized in terms of time compared to EM models. An EM simulator can be used for synthesis, which requires typically 10–15 iterations to generate inverse model dimensions. Comparisons of time to obtain the dimensions using the EM and the trained neural network models are listed in Table IV. It shows that the time required by the neural network inverse models are negligible compared to EM models.

F. Additional Discussions on the Examples

In this paper, the three-layer multilayer perceptron neural network structure was used for each neural network model, and a quasi-Newton training algorithm was used to train the neural network models. Testing data are used after training the model to verify the generalization ability of these models. Automatic model generation algorithm of NeuroModelerPlus [25] was used to develop these models, which automatically train the model until model training and testing accuracy are satisfied. The training error and test errors are generally similar because sufficient training data was used in the examples.

The coupling value in this study is formulated as coupling bandwidth since they are the product of normalized coupling values and bandwidth. In this way, bandwidth is no longer needed as a model input, which helps in reducing training data and increasing model accuracy.

The tuning time is approximately the same for both the EM and neural network design. Even though the EM method gives the best solution of a filter, the physical machining process cannot guarantee 100% accurate dimension. Therefore, after manufacturing, the filter tuning is required. The amount of time spent on tuning also depends on how accurate the dimensions are. If the dimensions are far different from their perfect values, then tuning time will increase. The neural network method provides approximately the same dimension as the EM method. They both provide excellent starting points for tuning. As a result, the tuning time is relatively short and is the same for both the EM and neural network methods. Consequently, the tuning time does not alter the comparison between the EM and neural network method.

The training time for the direct inverse tuning screw model is approximately 6 min. In the proposed algorithm, if we perform segmentation, it will add 28.5 s, and if multivalued solutions are detected in a segment, it adds another 7.5 s for a forward model for a small segment containing 200 samples. The training time for the complete inverse tuning screw model using the proposed methodology is approximately 5.5 h. For the coupling iris, the direct inverse model containing 37000 samples takes 26 min to train. The proposed method divides the model into four smaller segments, each containing approximately 9000 samples, and takes ten additional minutes per segment. The time to train a direct IO iris inverse model containing 125 000 data requires 2.5 h. The training time using the proposed methodology is 6 h, including the time for training segmented models. The training time for these models were obtained using the NeuroModelerPlus parallel-automated model generation algorithm [25] on an Intel Quad core processor at 2.4 GHz. The training time for the proposed inverse models is longer than that of the direct inverse models. However, once the models are trained, the proposed model is very fast for the designer, providing solutions nearly instantly.

The technique is useful for highly repeated design tasks such as designing filters of different orders and different specifications. The technique is not suitable if the inverse model is for the purpose of only one or a few particular designs because the model training time will make the technique cost ineffective. Therefore, the technique should be applied to inverse tasks, which will be reused frequently. In such a case, the benefit of using the models far outweigh the cost of training because of the following four reasons.

1) Training is a one-time investment, and the benefit of the model increases when the model is used over and over again. For example, the two different filters in this paper use the same set of iris and tuning screw models.
2) Conventional EM design is part of the design cycle, while neural network training is outside the design cycle.
3) Circuit design requires much human involvement, while neural network training is a machine-based computational task.
4) Neural network training can be done by a model developer and the trained model can be used by multiple designers.

The neural network approach cuts expensive design time by shifting much burden to offline computer-based neural network training [2]. An even more significant benefit of the proposed technique is the new feasibility of interactive design and what–if analysis using the instant solutions of inverse neural networks, substantially enhancing design flexibility and efficiency.

V. Conclusion

Efficient neural network modeling techniques have been presented and applied to microwave filter modeling and design. The inverse modeling technique has been formulated and the nonuniqueness of the IO relationship has been addressed. Methods to identify multivalued solutions and divide training data have been proposed for training inverse models. Data of the inverse model have been divided based on derivatives of the forward model and then used separately to train more accurate inverse sub-models. A method to correctly combine the inverse sub-models has been presented. The inverse models developed using the proposed techniques are more accurate than that using the direct method. The proposed methodology has been applied to waveguide filter modeling. Very good correlation was found between neural network predicted dimensions and that of perfectly tuned filters. This modeling approach is useful for a fast solution to inverse problems in microwave design.

References

[1] Q. J. Zhang, K. C. Gupta, and V. K. Devabhaktuni, "Artificial neural networks for RF and microwave design—From theory to practice," *IEEE Trans. Microw. Theory Tech.*, vol. 51, no. 4, pp. 1339–1350, Apr. 2003.

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

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

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

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

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

[7] B. Davis, C. White, M. A. Reece, M. E. Bayne, Jr., W. L. Thompson, II, N. L. Richardson, and L. Walker, Jr., "Dynamically configurable pHEMT model using neural networks for CAD," in *IEEE MTT-S Int. Microw. Symp. Dig.*, Philadelphia, PA, Jun. 2003, vol. 1, pp. 177–180.

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

[9] V. Markovic, Z. Marinkovic, and N. Males-llic, "Application of neural networks in microwave FET transistor noise modeling," in *Proc. Neural Network Applicat. Elect. Eng.*, Belgrade, Yugoslavia, Sep. 2000, pp. 146–151.

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

[11] J. P. Garcia, F. Q. Pereira, D. C. Rebenaque, J. L. G. Tornero, and A. A. Melcon, "A neural-network method for the analysis of multilayered shielded microwave circuits," *IEEE Trans. Microw. Theory Tech.*, vol. 54, no. 1, pp. 309–320, Jan. 2006.

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

[13] M. M. Vai, S. Wu, B. Li, and S. Prasad, "Reverse modeling of microwave circuits with bidirectional neural network models," *IEEE Trans. Microw. Theory Tech.*, vol. 46, no. 10, pp. 1492–1494, Oct. 1998.

[14] S. Selleri, S. Manetti, and G. Pelosi, "Neural network applications in microwave device design," *Int. J. RF Microw. Comput.-Aided Eng.*, vol. 12, pp. 90–97, Jan. 2002.

[15] J. W. Bandler, M. A. Ismail, J. E. Rayas-Sanchez, and Q. J. Zhang, "Neural inverse space mapping (NISM) optimization for EM-base microwave design," *Int. J. RF Microw. Comput.-Aided Eng.*, vol. 13, pp. 136–147, Mar. 2003.

[16] 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 Microw. Comput.-Aided Eng.*, vol. 9, pp. 287–296, May 1999.

[17] J. M. Cid and J. Zapata, "CAD of rectangular-waveguide H-plane circuits by segmentation, finite elements and artificial neural networks," *Electron. Lett.*, vol. 37, pp. 98–99, Jan. 2001.

[18] A. Mediavilla, A. Tazon, J. A. Pereda, M. Lazaro, I. Santamaria, and C. Pantaleon, "Neuronal architecture for waveguide inductive iris bandpass filter optimization," in *Proc. IEEE INNS–ENNS Int. Neural Networks Joint Conf.*, Como, Italy, Jul. 2000, vol. 4, pp. 395–399.

[19] F. Nunez and A. K. Skrivervik, "Filter approximation by RBF-NN and segmentation method," in *IEEE MTT-S Int. Microw. Symp. Dig.*, Fort Worth, TX, Jun. 2004, vol. 3, pp. 1561–1564.

[20] P. Burrascano, M. Dionigi, C. Fancelli, and M. Mongiardo, "A neural network model for CAD and optimization of microwave filters," in *IEEE MTT-S Int. Microw. Symp. Dig.*, Baltimore, MD, Jun. 1998, vol. 1, pp. 13–16.

[21] A. S. Ciminski, "Artificial neural networks modeling for computer-aided design of microwave filter," in *Int. Microw., Radar, Wireless Commun. Conf.*, Gdansk, Poland, May 2002, vol. 1, pp. 96–99.

[22] C. Kudsia, R. Cameron, and W.-C. Tang, "Innovations in microwave filters and multiplexing networks for communications satellite systems," *IEEE Trans. Microw. Theory Tech.*, vol. 40, no. 6, pp. 1133–1149, Jun. 1992.

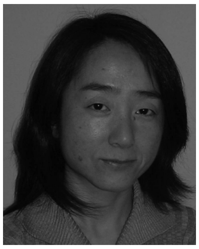
[23] Y. Wang, M. Yu, H. Kabir, and Q. J. Zhang, "Effective design of cross-coupled filter using neural networks and coupling matrix," in *IEEE MTT-S Int. Microw. Symp. Dig.*, San Francisco, CA, Jun. 2006, pp. 1431–1434.

[24] I. Bahl*, Lumped Elements for RF and Microwave Circuits*. Boston, MA: Artech House, 2003.

[25] Q. J. Zhang, NeuroModelerPlus. Dept. Electron., Carleton Univ., Ottawa, ON, Canada.

Humayun Kabir (S'06) received the B.Sc. degree in electrical and electronic engineering from the Bangladesh University of Engineering and Technology, Dhaka, Bangladesh, in 1999, the M.S.E.E. degree in electrical engineering from the University of Arkansas, Fayetteville, in 2003, and is currently working toward the Ph.D. degree in electronics engineering at Carleton University, Ottawa, ON, Canada.

He was a Research Assistant with the High Density Electronics Center (HiDEC), University of Arkansas, Fayetteville, and COM DEV, Cambridge, ON, Canada, as a co-op student. He is currently a Teaching and Research Assistant with Carleton University. His research interests include RF/microwave design and optimization.



Ying Wang (M'05) received the B.Eng. and Masters degrees in electronic engineering from the Nanjing University of Science and Technology, Nanjing, China, in 1993 and 1996, respectively, and the Ph.D. degree in electrical engineering from the University of Waterloo, Waterloo, ON, Canada, in 2000.

From 2000 to 2007, she was with COM DEV, Cambridge, ON, Canada, where she was involved in the development of computer-aided design (CAD) software for design, simulation, and optimization of microwave circuits for space application. In 2007, she joined the Faculty of Engineering and Applied Sciences, University of Ontario Institute of Technology, Oshawa, ON, Canada, where she is currently an Assistant Professor. Her research interests include RF/microwave CAD, microwave circuits design, and radio wave propagation modeling.

Ming Yu (S'90–M'93–SM'01) received the Ph.D. degree in electrical engineering from the University of Victoria, Victoria, BC, Canada, in 1995.

In 1993, while working on his doctoral dissertation part time, he joined COM DEV, Cambridge, ON, Canada, as a Member of Technical Staff, where he was involved in the design of passive microwave/RF hardware from 300 MHz to 60 GHz. He was also a principal developer of a variety of COM DEV's design and tuning software for microwave filters and multiplexers. His varied experience with COM DEV also includes being the Manager of filter/multiplexer technology (Space Group) and Staff Scientist of corporate research and development (R&D). He is currently the Chief Scientist and Director of R&D. He is responsible for overseeing the development of RF microelectromechanical system (MEMS) technology, computer-aided tuning and EM modeling, and optimization of microwave filters/multiplexers for wireless applications. He is also an Adjunct Professor with the University of Waterloo, Waterloo, ON, Canada. He has authored or coauthored over 70 publications and numerous proprietary reports. He is frequently a reviewer for IEE publications. He holds eight patents with four pending.

Dr. Yu is vice chair of the IEEE Technical Coordinating Committee 8 (TCC, MTT-8) and is a frequent reviewer of numerous IEEE publications. Since 2006, he has been the chair of the IEEE Technical Program Committee 11 (TPC-11). He was the recipient of the 1995 and 2006 COM DEV Ltd. Achievement Award for the development of computer-aided tuning algorithms and systems for microwave filters and multiplexers.

Qi-Jun Zhang (S'84–M'87–SM'95–F'06) received the B.Eng. degree from the East China Engineering Institute, Nanjing, China, in 1982, and the Ph.D. degree in electrical engineering from McMaster University, Hamilton, ON, Canada, in 1987.

From 1982 to 1983, he was with the System Engineering Institute, Tianjin University, Tianjin, China. From 1988 to 1990, he was with Optimization Systems Associates (OSA) Inc., Dundas, ON, Canada, where he developed advanced microwave optimization software. In 1990, he joined the Department of Electronics, Carleton University, Ottawa, ON, Canada, where he is currently a Professor. He has authored over 180 publications. He authored *Neural Networks for RF and Microwave Design* (Artech House, 2000) and coedited *Modeling and Simulation of High-Speed VLSI Interconnects* (Kluwer, 1994). He contributed to *Encyclopedia of RF and Microwave Engineering*, (Wiley, 2005), *Fundamentals of Nonlinear Behavioral Modeling for RF and Microwave Design* (Artech House, 2005), and *Analog Methods for Computer-Aided Analysis and Diagnosis* (Marcel Dekker, 1988). He was a Guest Co-Editor for the "Special Issue on High-Speed VLSI Interconnects" of the *International Journal of Analog Integrated Circuits and Signal Processing* (Kluwer, 1994) and was a two-time Guest Editor for the "Special Issue on Applications of ANN to RF and Microwave Design" of the *International Journal of RF and Microwave CAE* (Wiley, 1999, 2002). He is an Editorial Board member of the *International Journal of RF and Microwave Computer-Aided Engineering* and the *International Journal of Numerical Modeling*. His research interests are neural network and optimization methods for high-speed/high-frequency circuit design.

Dr. Zhang is a member on the Editorial Board of the IEEE TRANSACTIONS ON MICROWAVE THEORY AND TECHNIQUES. He is a member of the Technical Committee on Computer-Aided Design (MTT-1) of the IEEE Microwave Theory and Techniques Society (IEEE MTT-S).