Parametric Modeling of Microwave Passive Components Using Sensitivity-Analysis-Based Adjoint Neural-Network Technique

Sayed Alireza Sadrossadat, *Student Member, IEEE*, Yazi Cao, *Member, IEEE*, and Qi-Jun Zhang, *Fellow, IEEE*

***Abstract*—This paper presents a novel sensitivity-analysis-based adjoint neural-network (SAANN) technique to develop parametric models of microwave passive components. This technique allows robust parametric model development by learning not only the input–output behavior of the modeling problem, but also derivatives obtained from electromagnetic (EM) sensitivity analysis. A novel derivation is introduced to allow complicated high-order derivatives to be computed by a simple artificial neural-network (ANN) forward-back propagation procedure. New formulations are deduced for exact second-order sensitivity analysis of general multilayer neural-network structures with any numbers of layers and hidden neurons. Compared to our previous work on adjoint neural networks, the proposed SAANN is easier to implement into an existing ANN structure. The proposed technique allows us to obtain accurate and parametric models with less training data. Another benefit of this technique is that the trained model can accurately predict derivatives to geometrical or material parameters, regardless of whether or not these parameters are accommodated as sensitivity variables in EM simulators. Once trained, the SAANN models provide accurate and fast prediction of EM responses and derivatives used for high-level optimization with geometrical or material parameters as design variables. Three examples including parametric modeling of coupled-line filters, cavity filters, and junctions are presented to demonstrate the validity of this technique.**

***Index Terms*—Modeling, neural networks, parametric modeling, passive components, sensitivity analysis.**

I. Introduction

ARTIFICIAL neural network (ANN) techniques have been recognized for modeling and optimization of microwave components and circuits in electromagnetic (EM)-based microwave design [1]–[5]. Design optimization often requires repetitive adjustments of the values of geometrical or material parameters and can be very time consuming. An ANN can learn EM responses as a function of geometrical variables through an automated training process, and the trained ANN model can be subsequently implemented in high-level circuit and system designs, allowing fast simulation and optimization [6]. To improve learning and generalization in ANNs, knowledge-based neural network approaches that incorporate prior knowledge, such as analytical expressions [7], empirical models [8], [9], or equivalent circuits [10], [11], into the model structure were developed. Using these techniques, accurate models can be built with less hidden neurons and trained with less data, therefore speeding up model development.

An advanced study, namely, the adjoint neural network technique, was carried out to model the behavior of large-signal field-effect transistor (FET) devices [12]. The adjoint neural network bridges the gap between large-signal FET charge model and capacitance training data, and between large-signal model and small-signal S-parameter data. It was an early attempt allowing exact sensitivity to be calculated in a neural-network model for microwave design.

Recent advances in EM simulation have led to the availability of sensitivity information in addition to EM simulations, such as [13]–[16]. An algorithm for efficient estimation of S-parameter sensitivities with the time-domain transmission-line modeling (TLM) method has been proposed in [17]. A time-domain algorithm for wideband adjoint variable method (AVM) sensitivity analysis for dispersive materials is presented in [18]. An adjoint-sensitivity-based topology optimization method for the design of patch antennas is developed in [19]. A self-adjoint-sensitivity analysis-based approach for enhancing the bandwidth of narrowband antennas is introduced in [20]. An algorithm for accelerating the space-mapping optimization using adjoint sensitivities is also shown in [21]. Here, we propose to exploit such sensitivity information to further enhance the efficiency and accuracy of ANN models for microwave passive components. In order to train ANN models to learn EM sensitivities, we need to use ANN outputs to represent these sensitivities. Furthermore, in order to train the sensitivity-based ANN models, we need the derivatives of sensitivity outputs, therefore leading to the need of both first- and second-order derivatives in ANN. The subject of ANN derivatives has been investigated in the ANN community, and several techniques for ANN sensitivity computation have been used to train ANN models, such as [22]–[25]. The most widely used method in ANN area was the back propagation method, which was one of the key milestones propelling the ANN research into the mainstream in the 1980s [22]. The back-propagation method used a systematic mechanism to propagate ANN training error starting from the output layer and down to the input layer. Through this process, the first-order derivatives of ANN outputs versus inputs

Manuscript received November 06, 2012; revised March 01, 2013; accepted March 06, 2013. Date of publication April 04, 2013; date of current version May 02, 2013. This work was supported by the National Sciences and Engineering Research Council (NSERC) of Canada.

The authors are with the Department of Electronics, Carleton University, Ottawa, ON, Canada K1S5B6 (sadr@doe.carleton.ca; yazi.cao@hotmail.com; qjz@doe.carleton.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.2013.2253793

0018-9480/$31.00 © 2013 IEEE

are obtained efficiently [23]. Another interesting ANN derivative method, a generalized recursive least square method incorporating first-order derivatives for the ANN training, was developed to improve the generalization ability of ANN models while getting a compact structure [24]. Furthermore, an ANN and its extension of derivatives were applied to predict the radar cross section of a nonlinearly loaded antenna [25]. All these methods are based on first-order derivatives in ANNs. An interesting method for second-order derivative computation in an ANN was presented in [12], where a rather generic neural-network structure was assumed, including a knowledge-based neural-network structure.

In this paper, we propose a novel sensitivity-analysis-based adjoint neural-network (SAANN) technique, which allows robust parametric model development by learning not only the input–output behavior of the EM modeling problem, but also the derivatives from EM sensitivity analysis. To simultaneously learn input–output behavior and the derivative information, a novel derivation is introduced to allow complicated high-order derivatives to be computed by a simple ANN backward–forward propagation procedure that can be conveniently accommodated by the existing ANN. New formulations are deduced for general multilayer neural-network structures with any numbers of layers and hidden neurons. Compared to the previous work [12], the proposed SAANN technique is easier and simpler to implement into an existing ANN structure. The SAANN technique, which incorporates the derivative information into the model training process, can enhance the capability of learning and generalization of parametric models. This technique introduces a new way to reduce the amount of training data needed in the model training process while retaining model accuracy. This is beneficial because generation of training data from EM simulation or measurement is often the major expense of the model development process, and thus the SAANN technique makes model development faster. Another benefit of this technique is that the trained model can be used to predict the derivative information with respect to any inputs of the model (geometrical or material variables), no matter if they are accommodated as sensitivity variables in EM simulation or not. Once trained, the SAANN models provide accurate and fast prediction of the EM responses and their corresponding derivatives used for high-level design optimization with geometrical and material parameters as design variables. The validity of the proposed approach is confirmed by three parametric modeling examples involving coupled-line filters, cavity filters, and junctions.

II. Analysis and Incorporation of Derivative Information Into Model Training Process

We propose to use the EM derivative information to train ANN models for EM problems. Let $\boldsymbol{x}$ and $\boldsymbol{y}$ represent the inputs and outputs of the original EM problems, respectively. Consider two cases of ANN learning of EM problems, ANN 1 learning only the EM input–output relationship ($\boldsymbol{x}-\boldsymbol{y}$ relationship, e.g., geometry versus S-parameters in the original microwave modeling problems), and ANN2 learning not only the $\boldsymbol{x}-\boldsymbol{y}$ relationship, but also the $\mathrm{d}\boldsymbol{y}/\mathrm{d}\boldsymbol{x}$ to $\boldsymbol{x}$ relationship. We illustrate the learning using three training samples and two testing samples, as shown in Fig. 1. With the conventional approach, ANN 1 learns the three training samples well; however, the trained ANN is not accurate at testing points unless more training data are added. Our proposed approach is to train the ANN (i.e., ANN 2 in the figure) to learn not only the three training samples, but also the exact derivatives of $\mathrm{d}\boldsymbol{y}/\mathrm{d}\boldsymbol{x}$ at these three training samples. From this figure, the training error of the typical ANN 1 is small, but its testing error is quite large. However, by learning training samples and their exact derivatives simultaneously, the proposed ANN 2 can match well not only training samples, but also testing samples.

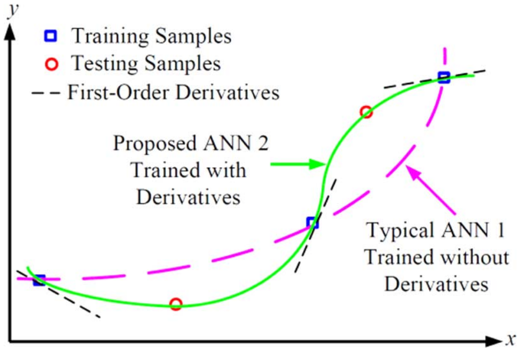


Fig. 1. Graphical illustration of ANN learning of $\boldsymbol{x}-\boldsymbol{y}$ relationship with or without using $\mathrm{d}\boldsymbol{y}/\mathrm{d}\boldsymbol{x}$ information. Trained without derivatives, the typical ANN 1 can obtain a small training error, but a larger testing error. Trained with derivatives, the new ANN 2 can obtain a small training error and a consistent testing error.

To further investigate such accuracy advantage of the proposed sensitivity training method, we use symbol $f_0(x)$ to represent the original $\boldsymbol{x}-\boldsymbol{y}$ relationship of the EM problems. Suppose that in theory $f_0(x)$ has continuous derivatives of any orders. Let x_0 be a training sample. Let $f_1(x)$ be the fitting curve by the conventional ANN approach (i.e., ANN 1 trained without using derivative data). Let $E_1(x_0)$ be the training error between $f_1(x)$ and $f_0(x)$ at training sample location x_0. Let $f_2(x)$ be the fitting curve by the proposed ANN (i.e., ANN 2 trained with derivative data). Let $E_2(x_0)$ and $E_2'(x_0)$ represent the training errors between $f_2(x)$ and $f_0(x)$ at x_0, and between ANN derivatives $f_2'(x)$ and derivative training data $f_0'(x)$ at x_0, respectively. Based on the Taylor expansion at x_0, the models $f_0(x)$, $f_1(x)$, and $f_2(x)$ can be expanded as

$$\begin{aligned} f_0(x) &= f_0(x_0)+f_0'(x_0)\cdot\Delta x+\sum_{i=2}^{n}\frac{1}{i!}f_0^{(i)}(x_0)\cdot\Delta x^i\\ f_1(x) &= f_0(x_0)+E_1(x_0)+f_1'(x_0)\cdot\Delta x+\sum_{i=2}^{n}\frac{1}{i!}f_1^{(i)}(x_0)\cdot\Delta x^i\\ f_2(x) &= f_0(x_0)+E_2(x_0)+f_0'(x_0)\cdot\Delta x+E_2'(x_0)\cdot\Delta x\\ &\quad+\sum_{i=2}^{n}\frac{1}{i!}f_2^{(i)}(x_0)\cdot\Delta x^i. \end{aligned} \tag{1}$$

In the ideal case, when the proposed and conventional ANNs are both trained very well, the training errors $E_1(x_0)$, $E_2(x_0)$, and $E_2'(x_0)$ will be all equal to 0. Assuming higher order parts of the equations are negligible because of small Δx^i, in such

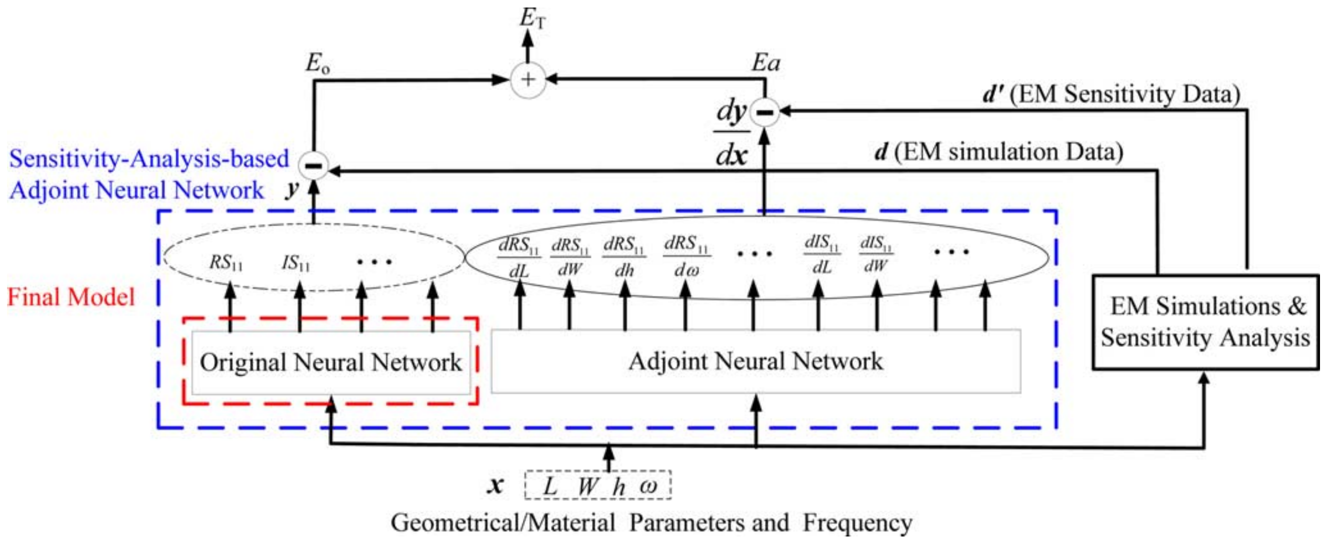


Fig. 2. Structure of the proposed SAANN model. It consists of two parts: original neural network and adjoint neural network, where L, W, h, and ω represent geometrical parameters such as length, width, and thickness of substrates and frequency, respectively.

an ideal case, the testing errors of proposed and conventional ANNs at testing sample $x = x_0 + \Delta x$ are

$$\begin{aligned} E_1(x_0+\Delta x) &= |f_1(x) - f_0(x)| \\ &= \Bigg| [f_1'(x_0) - f_0'(x_0)]\cdot \Delta x \\ &\quad + \sum_{i=2}^{n} \frac{1}{i!}\left[f_1^{(i)}(x_0) - f_0^{(i)}(x_0)\right]\cdot \Delta x^i \Bigg| \\ E_2(x_0+\Delta x) &= |f_2(x) - f_0(x)| \\ &= \left| \sum_{i=2}^{n} \frac{1}{i!}\left[f_2^{(i)}(x_0) - f_0^{(i)}(x_0)\right]\cdot \Delta x^i \right|. \end{aligned}$$

Clearly,

$$\lim_{\Delta x \to 0} \frac{E_2 - E_1}{\Delta x} = -|f_1'(x_0) - f_0'(x_0)| \le 0. \qquad (2)$$

Therefore, the testing error E_2 of the proposed ANN 2 is absolutely lower than the testing error E_1 of the conventional ANN 1 if the testing sample $x_0 + \Delta x$ is not far from the training sample x_0.

III. Proposed SAANN Technique

A. Structure of the Proposed SAANN Model

Let $\boldsymbol{x}$ be a vector representing the inputs of the original neural network such as frequency, geometrical, and material parameters of microwave passive components. Let $\boldsymbol{y}$ be a vector representing the outputs of the original neural network such as real and imaginary parts of S-parameters. Let $\boldsymbol{w}$ represent the synaptic weights of the original neural network. The adjoint neural network is a "companion" neural network sharing the same set of internal neuron-connection parameters as that in the original neural network, but with modified neuron activation functions such that the adjoint neural network provides first-order derivative information $\mathrm{d}\boldsymbol{y}/\mathrm{d}\boldsymbol{x}$. The detailed explanation of the adjoint neural-network concept is in [12].

The structure of the SAANN and its training is shown in Fig. 2. The SAANN model consists of two parts: the original neural network and the adjoint neural network. The inputs $\boldsymbol{x}$ of the SAANN model contain frequency, geometrical, and material parameters, which are the same as those of the original neural network. The outputs of the SAANN model contain the outputs $\boldsymbol{y}$ of the original neural network in addition to the derivatives $\mathrm{d}\boldsymbol{y}/\mathrm{d}\boldsymbol{x}$, which are the outputs of the adjoint neural network. Let $\boldsymbol{d}$ and $\boldsymbol{d}'$ be vectors representing the outputs of EM simulations (e.g., S-parameters) and the derivatives of S-parameters with respect to geometrical or material variables from EM sensitivity analysis, respectively. The object of the SAANN training is to adjust the internal weights $\boldsymbol{w}$ such that, for all training samples, the error between $\boldsymbol{y}$ and the training data $\boldsymbol{d}$ and $\mathrm{d}\boldsymbol{y}/\mathrm{d}\boldsymbol{x}$ and $\boldsymbol{d}'$ are minimized. Although the whole training process involves both the original and adjoint neural networks, the final parametric model can be fairly simple, only containing the original neural network, as shown in Fig. 2. Let the total training error be defined as

$$\begin{aligned} E_T &= E_o + E_a \\ &= \frac{1}{2}A\sum_{q\in Q}(y_q - d_q)^2 + \frac{1}{2}\sum_{p\in P, q\in Q} B_{q,p}\left(\frac{\partial y_q}{\partial x_p} - d'_{q,p}\right)^2 \end{aligned} \qquad (3)$$

where E_o and E_a represent the training error from original and adjoint neural-network models, respectively, x_p and y_q denote the pth input in $\boldsymbol{x}$ and the qth output in $\boldsymbol{y}$, respectively. P and Q represent index sets of inputs and outputs, respectively. $d'_{q,p}$ represent the training data for the derivative of the qth output with respect to the pth input. A and $B_{q,p}$ are the weighting factors for different terms in the error function (3), e.g., A representing the inverse of the minimum to maximum range of training data d_q, for $q \in Q$, and $B_{q,p}$ representing the inverse of the minimum to maximum range of training data $d'_{q,p}$, for $p \in P, q \in Q$.

B. Second-Order Derivatives for Training the SAANN Model

During the traditional ANN training process, only first-order derivatives are required to guide the gradient-based training process, and such first-order derivatives can be computed

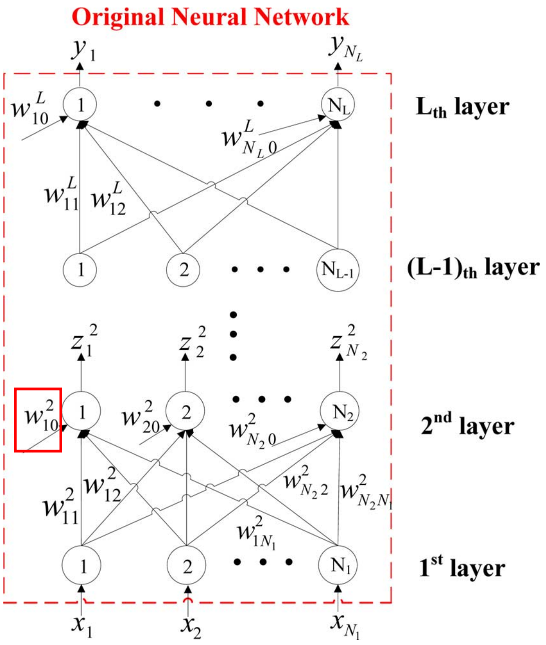

Original Neural Network
y_1
y_{N_L}
w_{10}^L
$w_{N_L0}^L$
w_{11}^L w_{12}^L
$\mathrm{L_{th}}$ layer
(L-1)$_{\mathrm{th}}$ layer
z_1^2 z_2^2 $z_{N_2}^2$
w_{10}^2 w_{20}^2 $w_{N_20}^2$
2$^{\mathrm{nd}}$ layer
w_{11}^2 w_{12}^2 $w_{N_22}^2$ $w_{N_2N_1}^2$ $w_{1N_1}^2$
1$^{\mathrm{st}}$ layer
x_1 x_2 x_{N_1}
1 2 $\mathrm{N_L}$ $\mathrm{N_{L-1}}$ $\mathrm{N_2}$ $\mathrm{N_1}$

Fig. 3. Structure of the original neural network.

through the back propagation method [6]. In order to train the original and adjoint neural network efficiently and simultaneously, the second-order derivatives with respect to ANN internal weights $\boldsymbol{w}$ should also be found.

The structure of the original neural network, as shown in Fig. 3, contains multilayers with the sigmoid function as the activation function in each hidden neuron.

Different from our previous work [12] where the second-order derivatives were calculated through a special computation process different from the original ANN, here a novel derivation is introduced to allow complicated second-order derivatives to be computed by a simpler ANN forward–backward propagation procedure, which can be conveniently accommodated by the existing ANN computational mechanism.

The proposed forward–backward propagation method is a combination of the standard back propagation procedure and a new procedure that maximally utilizes the ANN feedforward infrastructure already existing in typical ANN computations. The outputs of the ith hidden neuron in the lth layer of a standard multilayer perceptron (MLP) neural network are defined as [6]

$$z_i^l = \begin{cases} \gamma_i^l, & \text{for } i=1,2,\ldots,N_l; \quad l=L \\ \sigma\left(\gamma_i^l\right), & \text{for } i=1,2,\ldots,N_l; \quad l=2,\ldots,L-1 \\ x_i, & l=1 \end{cases} \tag{4}$$

where

$$\gamma_i^l = \sum_{\boxed{k=0}}^{N_{l-1}} w_{ik}^l \cdot z_k^{l-1}, \qquad i=1,2,\ldots,N_l; \quad l=2,3,\ldots,L \tag{5}$$

and w_{ik}^l is the weight between the ith hidden neuron of the lth layer and the kth hidden neuron of the $(l-1)$th layer, y_i is the ith output of the original neural network, $\sigma(\gamma)$ is the sigmoid function, N_l is the total number of hidden neurons in the lth layer, and L is the total number of layers. Note that for simplicity of the bias calculation, the first neuron in each layer is supposed to be 1, i.e., $z_0^l = 1$ $(l=1,2,\ldots,L)$.

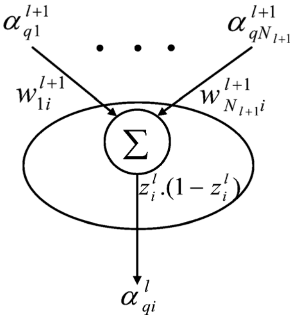


Fig. 4. Calculation of the proposed parameter α using the back propagation procedure available from the standard ANN procedure.

To calculate the second-order derivatives efficiently, we divide the derivatives of outputs $\boldsymbol{y}$ with respect to inputs $\boldsymbol{x}$ into two parts, as shown in (6), and define new variables of α_{qi}^l and β_{ip}^l as

$$\alpha_{qi}^l = \frac{\partial y_q}{\partial \gamma_i^l} \tag{6}$$

$$\beta_{ip}^l = \frac{\partial z_i^l}{\partial x_p} \tag{7}$$

where $l = 2,\ldots,L; q = 1,\ldots,N_L; i = 1,\ldots,N_l; p = 1,\ldots,N_1$

According to the definition of α_{qi}^l, for the last layer, i.e., $l = L$, α_{qi}^L is initialized as

$$\alpha_{qi}^L = \begin{cases} 1, & i=q \\ 0, & i\neq q \end{cases}; \quad i=1,2,\ldots,N_L; \quad q=1,2,\ldots,N_L. \tag{8}$$

α_{qi}^l can then be recursively calculated using the back propagation procedure

$$\boxed{\alpha_{qi}^l = \sum_{k=1}^{N_{l+1}} \frac{\partial y_q}{\partial \gamma_k^{l+1}} \cdot \frac{\partial \gamma_k^{l+1}}{\partial \gamma_i^l} = z_i^l \cdot \left(1 - z_i^l\right) \sum_{k=1}^{N_{l+1}} \alpha_{qk}^{l+1} \cdot w_{ki}^{l+1}} \tag{9}$$

for $i = 1,\ldots,N_l; l = L-1,\ldots,2; q = 1,\ldots,N_L$. This process is further illustrated in Fig. 4.

Now the adjoint neural network can be built using the α's as shown in Fig. 5. The outputs of the adjoint neural network, i.e., the derivative of the outputs $\boldsymbol{y}$ of the original neural network to inputs $\boldsymbol{x}$ can be calculated as

$$\frac{\partial y_q}{\partial x_p} = \sum_{i=1}^{N_2} \frac{\partial y_q}{\partial \gamma_i^2} \cdot \frac{\partial \gamma_i^2}{\partial x_p} = \sum_{i=1}^{N_2} \alpha_{qi}^2 \cdot w_{ip}^2 \tag{10}$$

where $p = 1,\ldots,N_1, q = 1,\ldots,N_L$, γ_i^2 is γ_j^l in (5) at the second layer, and w_{ip}^2 is the weight between the ith hidden neuron of the second layer and the pth hidden neuron of the first layer. This process is the same as the standard back propagation procedure [6], except that the starting error vector for back-propagation is a binary vector defined by (8) for a fixed q. In this way, the proposed parameters α are obtained with the minimum change to the standard ANN implementation. In this

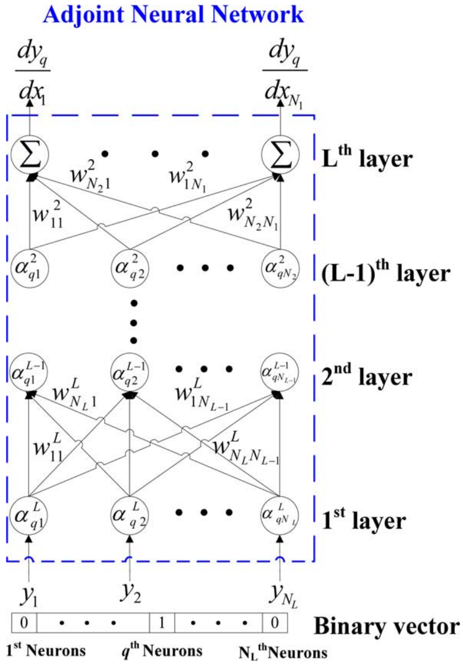


Fig. 5. Structure of the adjoint neural network (the outputs demonstrated here include only the derivatives of one of the outputs of the ANN to all of the inputs) using back propagation calculation of α^l_{qi} for each layer. As can be seen, the last layer computations contain only summation without an extra multiplication.

paper, an adjoint neural network is defined to represent the computation of the first-order derivatives in the original neural network. The adjoint neural network is illustrated in Fig. 5 (only derivatives of one of the outputs of the ANN to all the inputs are shown in this figure). The output of the adjoint neural network represents the derivative of the original neural network output with respect to the original neural network inputs. As seen in Fig. 5, the adjoint neural network is the reverse of the original neural network so that the number of inputs of the adjoint neural network is the number of outputs of the original neural network.

Next, we derive a simple method to compute β^l_{ip} for each layer by maximally utilizing the ANN feedforward infrastructure already existing in typical ANN computations. For each given index p, we formulate a systematic recursive procedure starting at the input layer. For the first layer, β^1_{ip} is initialized as

$$\beta^1_{ip} = \begin{cases} 1, & i = p \\ 0, & i \neq p \end{cases} \tag{11}$$

for $i = 1, \ldots, N_l; p = 1, \ldots, N_1$.

The next step using the feedforward procedure is to compute β^l_{ip} for the upper layers. According to definitions of z^l_i in (4) and γ^l_i in (5),

$$\begin{aligned} \beta^l_{ip} &= \frac{\partial z^l_i}{\partial \gamma^l_i} \cdot \frac{\partial \gamma^l_i}{\partial x_p} \\ &= z^l_i \cdot \left(1 - z^l_i\right) \cdot \frac{\partial \left(\sum_{k=0}^{N_{l-1}} w^l_{ik} \cdot z^{l-1}_k\right)}{\partial x_p} \\ &= z^l_i \cdot \left(1 - z^l_i\right) \cdot \sum_{k=1}^{N_{l-1}} w^l_{ik} \cdot \beta^{l-1}_{kp}, \qquad \begin{aligned} i &= 1, \ldots, N_l \\ l &= 2, \ldots, L-1. \end{aligned} \end{aligned} \tag{12}$$

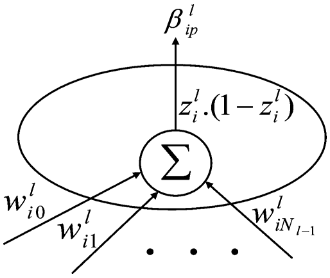


Fig. 6. Block diagram of β^l_{ip}. As shown in this figure, this block is very similar to a node in the original neural-network structure, except that the activation function in each neuron is a multiplication of $z^l_i(1 - z^l_i)$ instead of the sigmoid function.

According to the definitions of z^l_i in (4), z^l_i for the last layer is computed differently from other layers. Thus, the last step after calculating β^l_{ip} for all layers lower than L is to compute β^L_{ip} as

$$\beta^L_{ip} = \frac{\partial z^L_i}{\partial x_p} = \frac{\partial \gamma^L_i}{\partial x_p} = \sum_{k=0}^{N_{l-1}} w^L_{ik} \cdot \beta^{L-1}_{kp}, \qquad i = 1, \ldots, N_L. \tag{13}$$

Fig. 6 shows the inside of a typical β^l_{ip} block. It includes a multiplication after the summation. From this figure, we can see this block is very similar to a node in the original neural-network structure, except that the activation function in each neuron is a multiplication of $z^l_i(1 - z^l_i)$ instead of the sigmoid function.

Similar to the calculation of the first derivative information, there is another binary vector in the process of the calculation of β, but with the length of N_1 so that at the same time just one of the elements is 1 and it determines which x_p is selected for feedforward computation. Fig. 7 shows one step standard feedforward in the forward propagation method for calculating β.

Based on the calculation of α and β, the second-order derivatives can be obtained. We define

$$\theta^l_{qip} = \frac{\partial^2 y_q}{\partial \gamma^l_i \partial x_p}. \tag{14}$$

Firstly, for the output layer, i.e., the layer $l = L$, θ^L_{qip} needs to be initialized. According to the definition in (14), the first-order derivative of y_q to γ^L_i in the ith neuron in the output layer can be obtained as

$$\frac{\partial y_q}{\partial \gamma^L_i} = \begin{cases} 1, & q = i \\ 0, & q \neq i. \end{cases}$$

Since the above derivative is a constant value, its second-order derivative to input x_p is zero, i.e.,

$$\frac{\partial}{\partial x_p}\left(\frac{\partial^2 y_q}{\partial \gamma^L_i}\right) = 0.$$

Thus, for the output layer, θ^L_{qip} is initialized as

$$\theta^L_{qip} = \frac{\partial^2 y_q}{\partial \gamma^L_i \partial x_p} = 0 \tag{15}$$

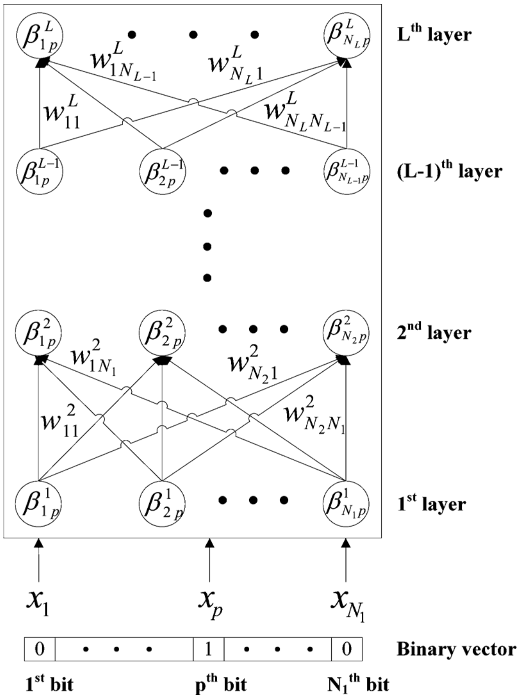


Fig. 7. One sample feedforward step in forward propagation method for the calculation of β for x_p. From this figure, we can see the calculation of β can be done within the original neural-network structure, except that the activation function is a multiplication of $z_i^l(1-z_i^l)$ instead of the sigmoid function.

for $q = 1, \ldots, N_L \quad i = 1, \ldots, N_L \quad p = 1, \ldots, N_1$. It indicates that for the output layer, the second-order derivatives of y_q to γ_i^L in the ith neuron and input x_p are fixed to zeros.

According to the definition of θ_{qip}^l in (14) for layers below the output layer, i.e., layer $l \neq L$,

$$\theta_{qip}^l = \frac{\partial\left(\frac{\partial y_q}{\partial \gamma_i^l}\right)}{\partial x_p} = \frac{\partial\left(\alpha_{qi}^l\right)}{\partial x_p}. \tag{16}$$

Utilizing (9), (16) now becomes

$$\theta_{qip}^l = \sum_{k=1}^{N_{l+1}} \left(\frac{\partial \alpha_{qk}^{l+1}}{\partial x_p} \cdot w_{kq}^{l+1} \cdot z_i^l \cdot \left(1 - z_i^l\right) + \frac{\partial\left(z_i^l \cdot \left(1 - z_i^l\right)\right)}{\partial x_p} \cdot \alpha_{qk}^{l+1} \cdot w_{kq}^{l+1} \right) \tag{17}$$

where utilizing the definition of β_{ip}^l in (7),

$$\frac{\partial\left(z_i^l \cdot \left(1 - z_i^l\right)\right)}{\partial x_p} = \left(1 - 2 \cdot z_i^l\right) \cdot \frac{\partial z_i^l}{\partial x_p} = \left(1 - 2 \cdot z_i^l\right) \cdot \beta_{ip}^l.$$

Therefore, θ_{qip}^l in (17) can be calculated recursively as

$$\theta_{qip}^l = z_i^l \cdot \left(1 - z_i^l\right) \cdot \sum_{k=1}^{N_{l+1}} \theta_{qkp}^{l+1} \cdot w_{kq}^{l+1} + \left(1 - 2 \cdot z_i^l\right) \cdot \beta_{ip}^l \cdot \sum_{k=1}^{N_{l+1}} \alpha_{qk}^{l+1} \cdot w_{kq}^{l+1} \tag{18}$$

for $l = L-1, \ldots, 2; \quad q = 1, \ldots, N_L; \quad i = 1, \ldots, N_l; \quad p = 1, \ldots, N_1$.

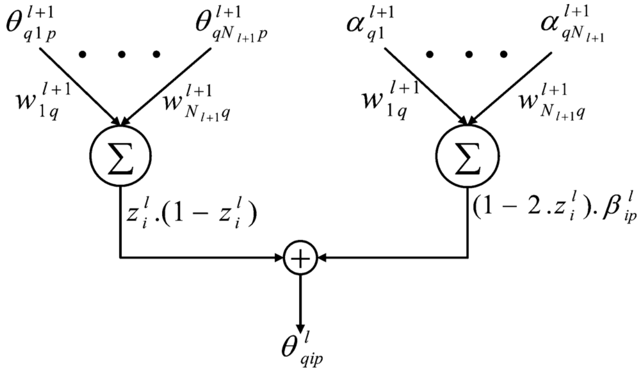


Fig. 8. Calculation of θ_{qip}^l using back propagation procedure. As shown in this figure, the calculation of θ_{qip}^l is very similar to the calculation of the first-order derivative information in addition to some extra multiplication factors.

Fig. 8 shows the calculation of θ_{qip}^l based on θ_{qip}^{l+1} and α_{qi}^{l+1} in the upper layer using a simple back propagation procedure. From this figure, we can see the calculation of θ_{qip}^l is quite similar to twice the standard back propagation calculation of the first-order derivatives in (9) in addition to two multiplications.

Now, the second-order derivatives of the outputs of the original neural network model, e.g., the derivatives of S-parameters to geometrical variables$\boldsymbol{x}$, to ANN internal weights $\boldsymbol{w}$ can be computed as

$$\frac{\partial^2 y_q}{\partial w_{ij}^l \cdot \partial x_p} = \frac{\partial\left(\frac{\partial y_q}{\partial \gamma_i^l} \cdot \frac{\partial \gamma_i^l}{\partial w_{ij}^l}\right)}{\partial x_p}. \tag{19}$$

According to (5),

$$\frac{\partial \gamma_i^l}{\partial w_{ij}^l} = z_j^{l-1}.$$

Equation (18) now becomes

$$\begin{aligned} \frac{\partial^2 y_q}{\partial w_{ij}^l \cdot \partial x_p} &= \frac{\partial^2 y_q}{\partial \gamma_i^l \partial x_p} \cdot \frac{\partial \gamma_i^l}{\partial w_{ij}^l} + \frac{\partial\left(\frac{\partial \gamma_i^l}{\partial w_{ij}^l}\right)}{\partial x_p} \frac{\partial y_q}{\partial \gamma_i^l} \\ &= \frac{\partial^2 y_q}{\partial \gamma_i^l \partial x_p} \cdot z_j^{l-1} + \frac{\partial z_j^{l-1}}{\partial x_p} \cdot \frac{\partial y_q}{\partial \gamma_i^l} \\ &= \theta_{qip}^l \cdot z_j^{l-1} + \beta_{jp}^{l-1} \cdot \alpha_{qi}^l. \end{aligned} \tag{20}$$

For $l = 2, \ldots, L; \quad q = 1, \ldots, N_L; \quad p = 1, \ldots, N_1; \quad i = 1, \ldots, N_l; \quad j = 1, \ldots, N_{l-1}$.

As shown in this (20), once α_{qi}^l, β_{ip}^{l-1}, and θ_{qip}^l are computed, the second-order derivatives of the outputs $\boldsymbol{y}$ to ANN internal weights $\boldsymbol{w}$ are readily obtained. Fig. 9 is a block diagram demonstrating the process of calculating of the second-order derivatives for the proposed SAANN model. To obtain the second-order derivatives, firstly β_{ip}^l has to be initialized following (11) with $l = 2$ and calculated recursively following (13) using the forward propagation procedure with increasing l untill$l = L$. α_{qi}^l is then initialized following (8) with $l = L$, and calculated recursively following (9) using the back propagation procedure with decreasing l until $l = 2$. Note that the calculation of α and β can be done in parallel. Next, θ_{qip}^l is initialized following (15) with $l = L$, and calculated recursively following (18) using

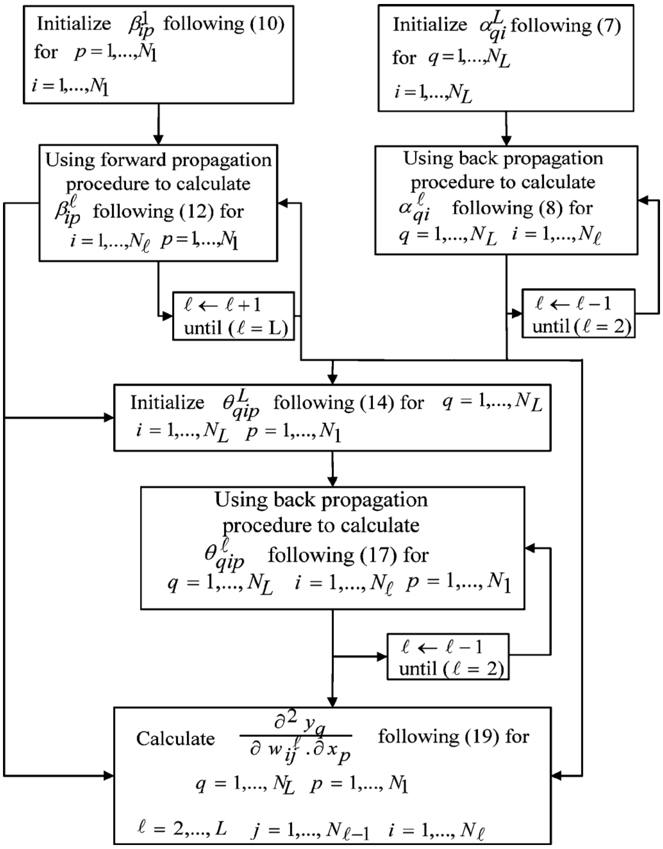

Initialize β_{ip}^{1} following (10) for $p=1,\dots,N_1$ $i=1,\dots,N_1$
Initialize α_{qi}^{L} following (7) for $q=1,\dots,N_L$ $i=1,\dots,N_L$
Using forward propagation procedure to calculate β_{ip}^{ℓ} following (12) for $i=1,\dots,N_\ell$ $p=1,\dots,N_1$
Using back propagation procedure to calculate α_{qi}^{ℓ} following (8) for $q=1,\dots,N_L$ $i=1,\dots,N_\ell$
$\ell \leftarrow \ell+1$ until (ℓ = L)
$\ell \leftarrow \ell-1$ until (ℓ = 2)
Initialize θ_{qip}^{L} following (14) for $q=1,\dots,N_L$ $i=1,\dots,N_L$ $p=1,\dots,N_1$
Using back propagation procedure to calculate θ_{qip}^{ℓ} following (17) for $q=1,\dots,N_L$ $i=1,\dots,N_\ell$ $p=1,\dots,N_1$
$\ell \leftarrow \ell-1$ until (ℓ = 2)
Calculate $\frac{\partial^2 y_q}{\partial w_{ij}^{\ell}\cdot\partial x_p}$ following (19) for $q=1,\dots,N_L$ $p=1,\dots,N_1$ $\ell=2,\dots,L$ $j=1,\dots,N_{\ell-1}$ $i=1,\dots,N_\ell$

Fig. 9. Calculation of the second-order derivatives of the proposed SAANN parametric model.

computed β_{ip}^{l} and α_{qi}^{l} and the back propagation procedure with decreasing l until $l = 2$. Finally, the computed α, β, and θ are used to calculate the second-order derivatives following (20).

IV. Application Examples

A. Parametric Modeling of a Coupled-Line Filter

In this example, we illustrate the use of the proposed SAANN technique to develop a parametric model for a family of coupled-line filters, as shown in Fig. 10, where S_1 and S_2 are the spacing between lines, and D_1, D_2, and D_3 are the offset distances from the ends of each coupled lines to the corresponding fringes, respectively.

The structure of the SAANN model for the coupled-line filter example is shown in Fig. 11. This parametric model has six inputs, i.e., $\boldsymbol{x} = [S_1\ \ S_2\ \ D_1\ \ D_2\ \ D_3\ \ \omega]^T$, which include five geometrical variables S_1, S_2, D_1, D_2, and D_3 defined in Fig. 10 and frequency ω. A 3-D EM simulator, i.e., CST Microwave Studio [13], is used to generate S-parameters and sensitivity information. In the implementation of sensitivity analysis in the EM simulator, the variables D_1, D_2, and D_3 are set as the sensitivity geometrical variables and the variables S_1 and S_2 are variables without sensitivity information. This SAANN model combining the original and adjoint neural networks used for training has 28 outputs, i.e., $[RS_{11}\ \ IS_{11}\ \ RS_{12}\ \ IS_{12}\ \ \frac{dRS_{11}}{dS_1}\ \ \frac{dRS_{11}}{dS_2}\ \ \frac{dRS_{11}}{dD_1}\ \ \frac{dRS_{11}}{dD_2}\ \ \frac{dRS_{11}}{dD_3}\ \ \frac{dRS_{11}}{d\omega}\ \ \frac{dIS_{11}}{dS_1}\ \ \frac{dIS_{11}}{dS_2}\ \ \frac{dIS_{11}}{dD_1}\ \ \frac{dIS_{11}}{dD_2}\ \ \frac{dIS_{11}}{dD_3}\ \ \frac{dIS_{11}}{d\omega}$, $\dots, \frac{dIS_{12}}{d\omega}]^T$, which are real and imaginary parts of S_{11} and S_{12}, the derivatives of real and imaginary parts of S_{11} and S_{12} with respect to six input variables (including frequency). The sensitivity analysis in the EM simulator is performed to obtain the derivatives of real and imaginary parts of S_{11} and S_{12} to three sensitivity variables D_1, D_2, and D_3. Since the other variables are not available from EM simulation (i.e., S_1, S_2, and ω are not sensitivity variables in CST EM simulation), the corresponding outputs from the SAANN model are left as free variables in the model training process. This is achieved by setting the training weights for $[\frac{dRS_{11}}{dS_1}\ \ \frac{dRS_{11}}{dS_2}\ \ \frac{dRS_{11}}{d\omega}\ \ \frac{dIS_{11}}{dS_1}\ \ \frac{dIS_{11}}{dS_2}\ \ \frac{dIS_{11}}{d\omega}\ \ \frac{dRS_{12}}{dS_1}\ \ \frac{dRS_{12}}{dS_2}\ \ \frac{dRS_{12}}{d\omega}\ \ \frac{dIS_{12}}{dS_1}\ \ \frac{dIS_{12}}{dS_2}\ \ \frac{dIS_{12}}{d\omega}]^T$ as 0 in our training program [26]. The frequency range is from 2 to 2.9 GHz with a step size of 2.7 MHz. In order to show the merits of the SAANN technique, which can enhance the capability of learning and generalization of the overall models with less training data, the data range of training data and testing data is defined in Table I. Partial orthogonal design of the experiments method [27] is used to determine the size of training and testing data. Although the whole training process involved the original neural network and adjoint neural network, the final parametric model is simple, only containing the original neural network.

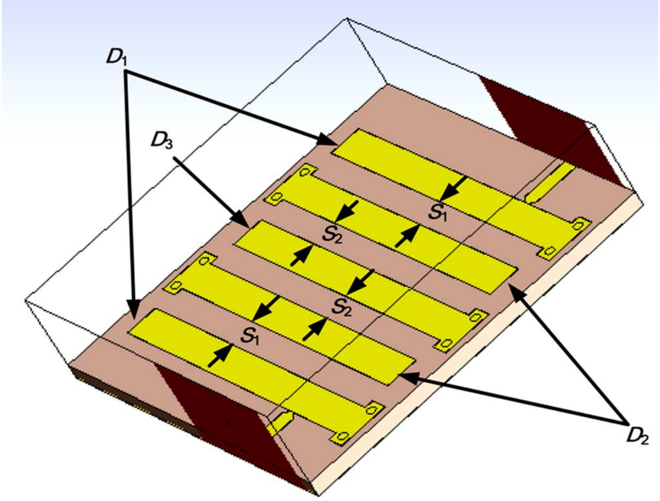


Fig. 10. Structure of a coupled-line filter and geometrical parameters used for generating training data for parametric modeling example.

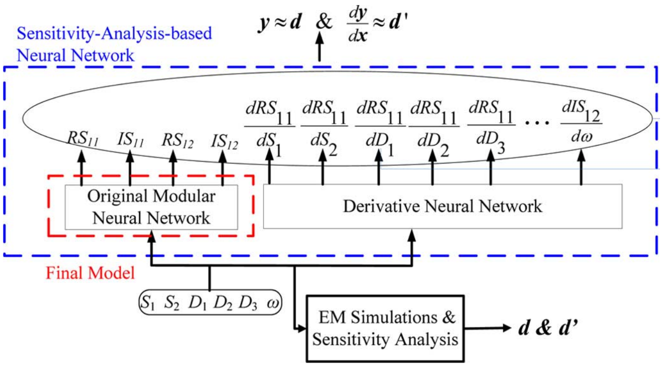

$y \approx d$ & $\frac{dy}{dx} \approx d'$
Sensitivity-Analysis-based Neural Network
RS_{11} IS_{11} RS_{12} IS_{12} $\frac{dRS_{11}}{dS_1}$ $\frac{dRS_{11}}{dS_2}$ $\frac{dRS_{11}}{dD_1}$ $\frac{dRS_{11}}{dD_2}$ $\frac{dRS_{11}}{dD_3}$ ... $\frac{dIS_{12}}{d\omega}$
Original Modular Neural Network
Derivative Neural Network
Final Model
S_1 S_2 D_1 D_2 D_3 ω
EM Simulations & Sensitivity Analysis
d & d'

Fig. 11. Structure of the parametric SAANN model for coupled-line filters.

Fig. 12 depicts the outputs of the proposed SAANN model for three different geometries #1, #2, and #3, and its compar-

TABLE I
DEFINITION OF TRAINING AND TESTING DATA FOR THE COUPLED-LINE FILTER EXAMPLE

Parameters		Training data			Testing data		
		Min	Max	Step	Min	Max	Step
S_1(mm)		36	44	1	36.5	43.5	1
S_2(mm)		36	44	1	36.5	43.5	1
Sensitivity Variables	D_1 (mm)	4	8	0.2	4.1	7.9	0.2
	D_2 (mm)	-4.6	-0.6	0.2	-4.5	-0.7	0.2
	D_3 (mm)	-4.4	-0.4	0.2	-4.3	-0.5	0.2

ison with EM data and a conventional ANN model trained with training data of different sizes.

The geometrical variables for three coupled-line filters are as follows.

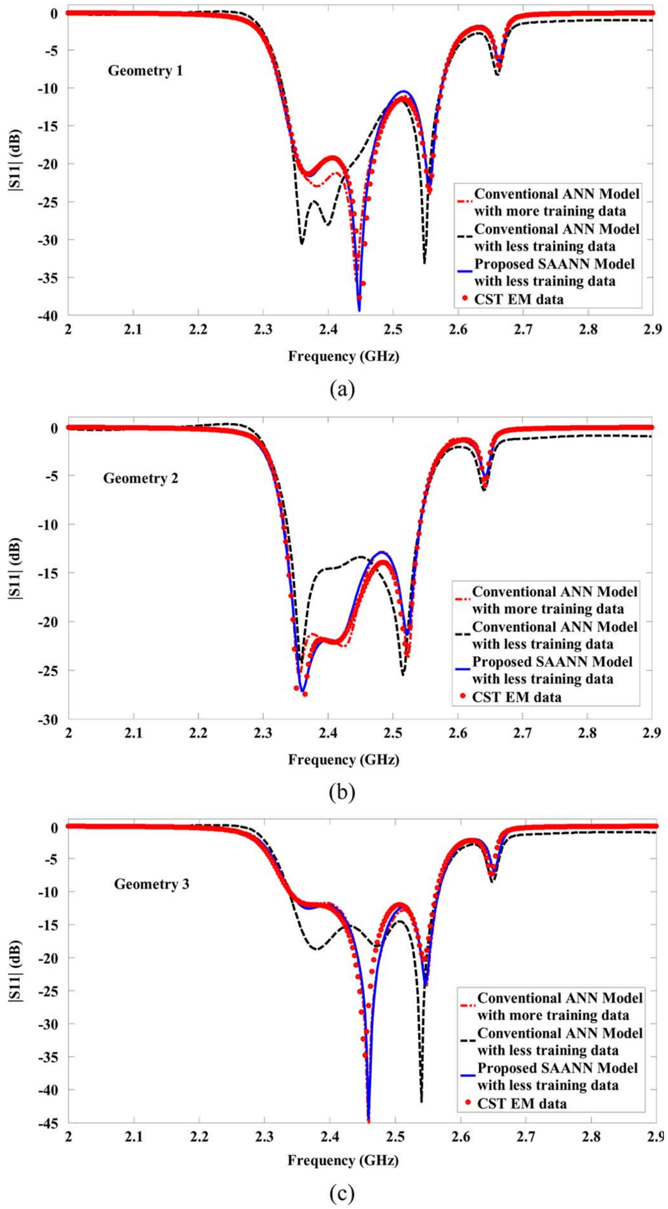
- Geometry 1: $S_1 = 39.5$ mm, $S_2 = 37.5$ mm, $D_1 = 4.1$ mm, $D_2 = -2.5$ mm, $D_3 = -2.3$ mm.
- Geometry 2: $S_1 = 40.5$ mm, $S_2 = 38.5$ mm, $D_1 = 7.5$ mm, $D_2 = -1.1$ mm, $D_3 = -1.1$ mm.
- Geometry 3: $S_1 = 36.5$ mm, $S_2 = 38.5$ mm, $D_1 = 6.5$ mm, $D_2 = -3.1$ mm, $D_3 = -0.5$ mm.

As shown in Fig. 12, broadband accuracy of the proposed SAANN model is confirmed by its good agreement with EM data in terms of S_{11} even though these geometries are never used in the training process.

As shown in Table II, an SAANN trained with few data can achieve similar accuracy as a conventional ANN trained with much more data. In this way, the development time for the proposed SAANN model is much shorter than that of conventional ANN. All simulations in this paper are done on the same computer with Intel core 2 quad CPU@2.4 GHz and 4-GB memory. The obtained ANN model achieves almost the same solutions as CST EM simulations using much less time. The SAANN model development cost for this coupled-line filter example, including training data generation time (40 sets of training geometries) and model training time, is about 5.46 h and for conventional ANN model development (120 sets of training geometries) is about 15.5 h. Note that the training is a one-time investment, and the benefit of using the model accumulates when the model is used over and over again.

Here, we show another benefit of this proposed SAANN technique that the trained model can accurately predict the derivative information with respect to geometrical variables. As shown in Fig. 13, we provide the comparison of the derivative information of the real part of S_{11} with respect to sensitivity variablesD_1, D_2, and D_3 by the proposed SAANN parametric model and CST sensitivity analysis at geometries #1, #2, and #3, respectively. This figure confirms that the proposed SAANN model can approximate the derivative information well, even though the geometry values have never been used in training.

In Fig. 14, we utilize the sensitivity ability of the SAANN to predict the derivative information of the real part of S_{11} with respect to nonsensitivity variables S_1 and S_2 by the SAANN parametric model and perturbation sensitivity at geometries #1, #2, and #3, respectively. This figure demonstrates that the SAANN parametric model can be used to accurately predict the derivative information with respect to geometrical variables, which can even be nonsensitivity variables in EM simulation.



(a)

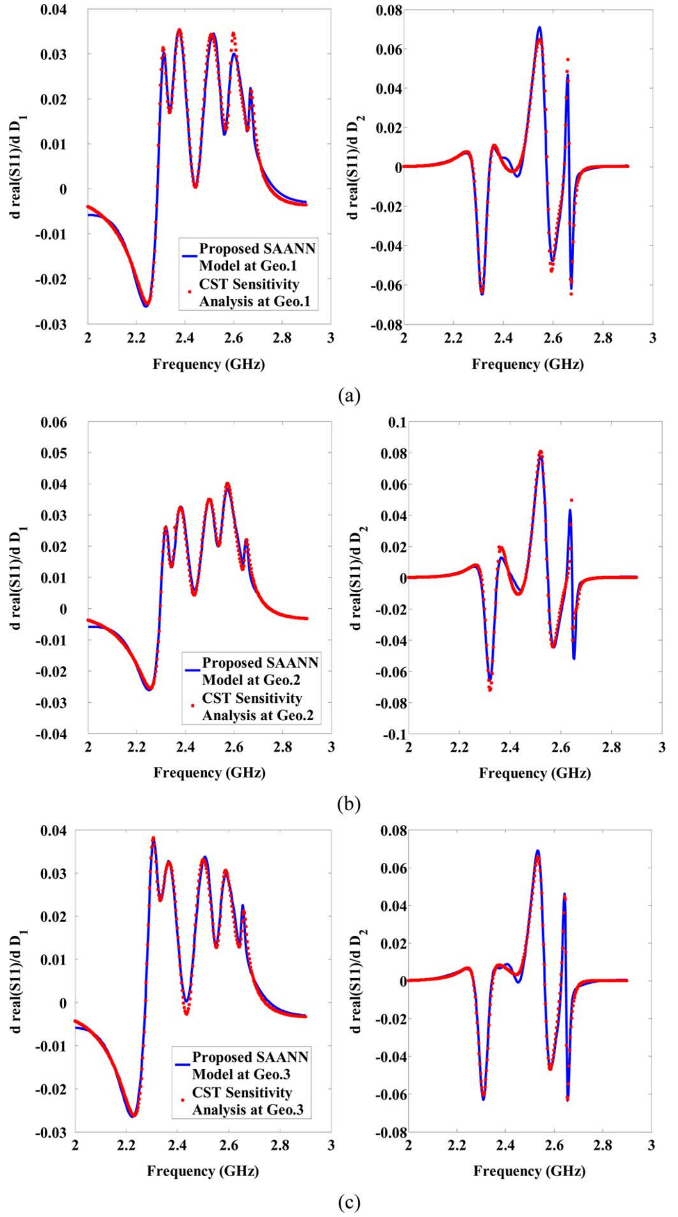
(b)

(c)

Fig. 12. Comparison of the magnitude in decibels of S_{11} of the SAANN model trained with less data (40 sets of data), CST EM data, conventional ANN model trained with less data (40 sets of data), and conventional ANN model trained with more data (120 sets of data) for three different filter geometries. These three different geometries are from test data and have never been used in training. As shown in this figure, using the SAANN model, we can use less training data to achieve good model accuracy than that needed for the conventional ANN model.

TABLE II
TRAINING AND TESTING RESULTS FOR COUPLED-LINE FILTER EXAMPLE

Model Type	Original Neural Network Structure	Average Training Error	Average Testing Error
Conventional ANN Model using 120 sets of training data	6-40-4	0.897%	0.989%
Conventional ANN Model using 40 sets of training data	6-35-4	1.073%	4.357%
Proposed SAANN Model using 40 sets of training data	6-35-4	0.871%	0.946%

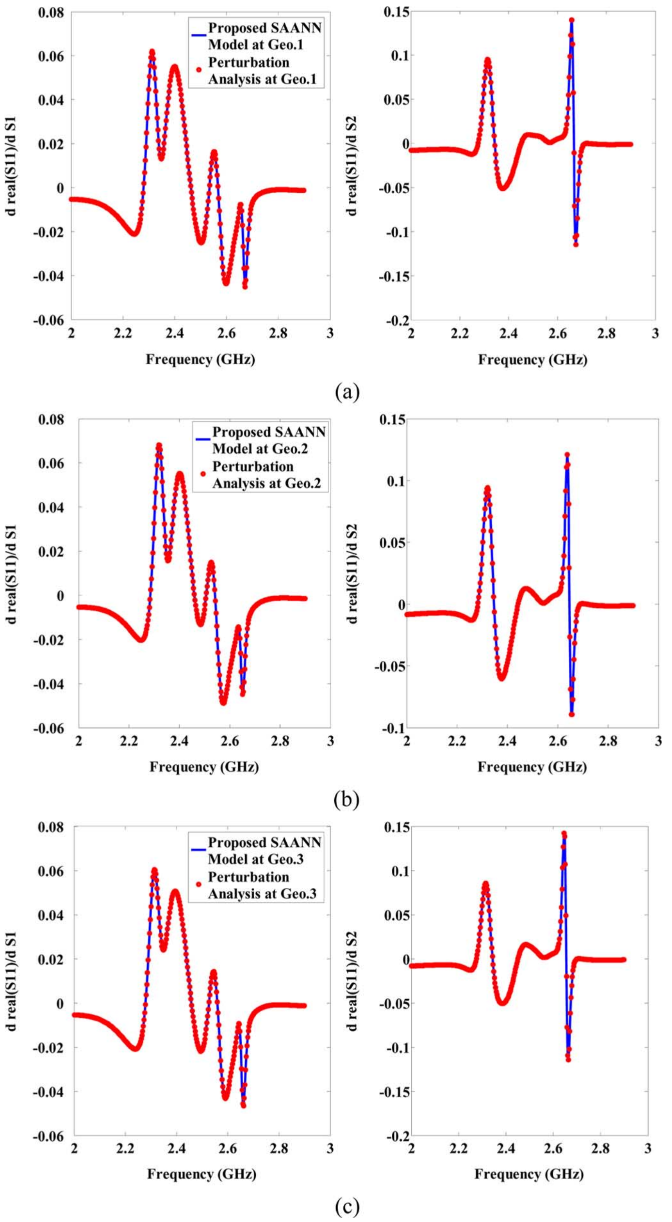


(a)

(b)

(c)

Fig. 13. Comparison of the derivative information of the real part of S_{11} to sensitivity variablesD_1, D_2, and D_3 by the proposed SAANN model and CST sensitivity analysis for $dRS_{11}/\mathrm{d}D_1$, and $dRS_{11}/\mathrm{d}D_2$ at geometries: (a) #1, (b) #2, and (c) #3 for the coupled-line filter example. As shown in this figure, the proposed SAANN model can accurately predict the derivative information, which are much closer to those obtained from CST sensitivity analysis, even though such geometries are never used in the training process.



(a)

(b)

(c)

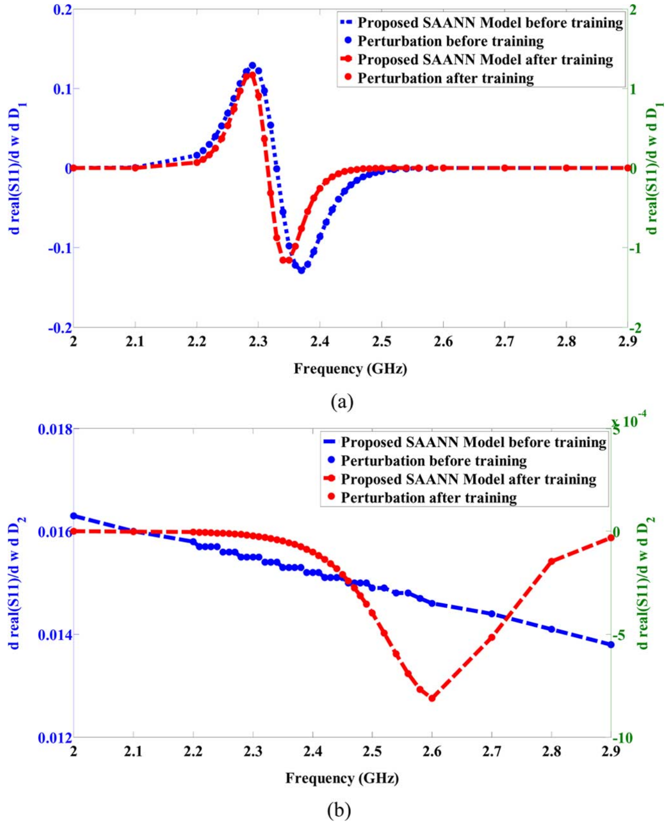
Fig. 14. Derivative information of the real part of S_{11} to nonsensitivity variables S_1 and S_2 by the proposed SAANN model and perturbation sensitivity for $dRS_{11}/\mathrm{d}S_1$ and $dRS_{11}/\mathrm{d}S_2$ at geometries: (a) #1, (b) #2, and (c) #3 for the coupled-line filter example. As shown in this figure, the proposed SAANN parametric model can predict the derivative information with respect to the geometrical variables even though these variables are not available as sensitivity variables in original EM simulation.

As an example to demonstrate the validity of the proposed second-order derivatives calculation in the SAANN technique, Fig. 15 compares the second-order derivatives of the real part of S_{11} to variables D_1 or D_2 and ANN weights w_{11}^2 and w_{11}^3 at geometry #1 by the SAANN parametric model versus that from perturbation as a continuous function in frequency sub-spaces before and after training, respectively. The good agreement in those figures verifies our proposed formulas (6)–(20) for the second-order derivatives calculation in the SAANN technique.

B. Parametric Modeling of a Junction

In this example, the proposed SAANN technique is applied to develop the parametric model of a family of junctions, as shown in Fig. 16, where g is the gap distance between two conductive walls, d_h is the height of the tuning cylinder, and d_r is the radius of the tuning cylinder.

The structure of the proposed SAANN parametric model for the junction example is shown in Fig. 17. This parametric model has four inputs, i.e., $\boldsymbol{x} = [g \ \ d_h \ \ d_r \ \ \omega]^T$, which include three geometrical variables g, d_h, and d_r defined in Fig. 16 and frequency ω. In this example, g, dh, and dr are all set as the sensitivity variables. This SAANN model combining the original and adjoint neural networks used for training has 40 outputs, i.e., $[RS_{11} \ IS_{11} \ RS_{21} \ IS_{21} \ RS_{31} \ IS_{31} \ RS_{41} \ IS_{41} \ \frac{dRS_{11}}{dg} \ \frac{dRS_{11}}{dd_h} \ \frac{dRS_{11}}{dd_r} \ \frac{dRS_{11}}{d\omega} \ \frac{dIS_{11}}{dg} \ \frac{dIS_{11}}{dd_h} \ \frac{dIS_{11}}{dd_r} \ \frac{dIS_{11}}{d\omega}, \ldots, \frac{dIS_{41}}{d\omega}]^T$, which are



(a)

(b)

Fig. 15. Comparison of second-order derivatives of the real part of S_{11} to variables D_1 or D_2 and ANN weights: (a) d^2 real $(S_{11})/\mathrm{d}w_{11}^2\mathrm{d}D_1$ and (b) d^2 real $(S_{11})/\mathrm{d}w_{11}^3\mathrm{d}D_2$ versus frequency at geometry #1 before and after ANN training. Good agreement is observed between the proposed SAANN technique and EM perturbation techniques regardless of whether the ANN is trained or not.

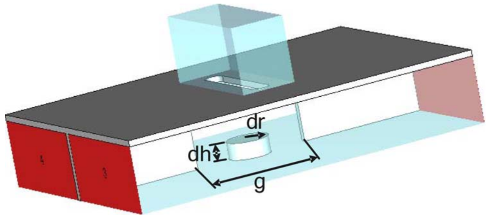


Fig. 16. Structure of a junction and geometrical parameters used for generating training data for parametric modeling example.

real and imaginary parts of S_{11} S_{21} S_{31}, and S_{41}, the derivatives of real and imaginary parts of S_{11} S_{21} S_{31} and S_{41} with respect to four input variables (including frequency). The sensitivity analysis in CST EM simulator is performed to obtain the derivatives of real and imaginary parts of S_{11} S_{21} S_{31} and S_{41} to three sensitivity variables. Since frequency ω is not a sensitivity variable in EM simulation, the corresponding outputs from the SAANN parametric model are left as free variables in the model training process. This is achieved by setting the training weights for $[\frac{dRS_{11}}{d\omega}\ \frac{dIS_{11}}{d\omega}\ \frac{dRS_{21}}{d\omega}\ \frac{dIS_{21}}{d\omega}\ \frac{dRS_{31}}{d\omega}\ \frac{dIS_{31}}{d\omega}\ \frac{dRS_{41}}{d\omega}\ \frac{dIS_{41}}{d\omega}]^T$ as 0 in our training program. The frequency range is from 7 to 9 GHz with a step size of 6 MHz. The data range of training data and testing data is defined in Table III. Partial orthogonal design of the experiments method is also used to determine training and testing data.

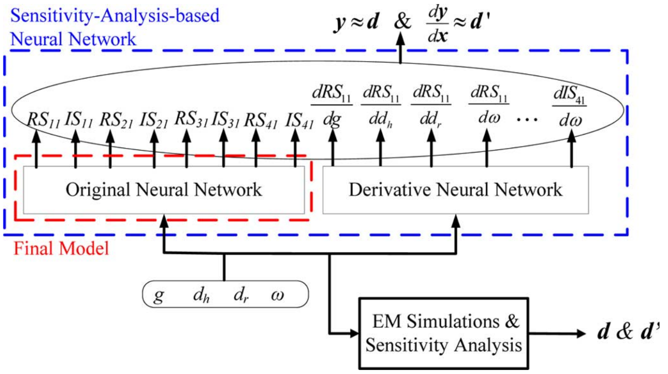

Sensitivity-Analysis-based
Neural Network
y ≈ d & dy/dx ≈ d'
RS11 IS11 RS21 IS21 RS31 IS31 RS41 IS41
dRS11/dg dRS11/ddh dRS11/ddr dRS11/dω ... dIS41/dω
Original Neural Network
Derivative Neural Network
Final Model
g dh dr ω
EM Simulations & Sensitivity Analysis
d & d'

Fig. 17. Structure of the proposed SAANN parametric model for the junction example.

TABLE III
DEFINITION OF TRAINING AND TESTING DATA FOR JUNCTION EXAMPLE

Parameters		Training data			Testing data		
		Min	Max	Step	Min	Max	Step
Sensitivity Variables	g (mm)	16	24	1	16.5	23.5	1
	d_h (mm)	1.5	3.5	0.2	1.6	3.4	0.2
	d_r (mm)	2	4	0.2	2.1	3.9	0.2

TABLE IV
TRAINING AND TESTING RESULTS FOR JUNCTION EXAMPLE

Model Type	Original Neural Network Structure	Average Training Error	Average Testing Error
Conventional ANN Model using 80 sets of training data	4-20-8	0.413%	0.473%
Conventional ANN Model using 15 sets of training data	4-20-8	0.482%	0.862%
Proposed SAANN Model using 15 sets of training data	4-20-8	0.453%	0.531%

Table IV shows final results of training in terms of the average training and testing error of the final trained model and its comparison with the conventional ANN model, which was trained without using EM derivative data. Two sets of training data were used in order to compare the effect of training with respect to different sizes of training data. One set of training data has 80 samples (i.e., training with more training data), and another set has only 15 samples (i.e., training with less training data). From this table, we can see that with more training data, the conventional ANN model (i.e., trained without sensitivity information) can obtain a small training error and a consistent testing error. With less training data, a conventional ANN model trained without sensitivity information can obtain a small training error, but a larger testing error since the limited training data could not adequately represent the whole EM behavior of the original modeling problem. In contrast, the proposed SAANN parametric model can obtain a small training error and a small testing error with the same number of training data using sensitivity information to training neural networks. This technique introduced a new way to decrease the necessary training data in the model training process.

The SAANN model development cost for this junction example, including training data generation time (15 sets of training data) and model training time, is about 9.4 h, and for the conventional ANN model development (80 sets of training geometries) is about 45.7 h. This further demonstrates that using the proposed technique, we speed up the model development time. Note that the training is a one-time investment, and the benefit of using the model accumulates when the model is used over and over again.

Fig. 18 depicts the outputs of the proposed SAANN parametric model for three different junction geometries #1, #2, and #3, and its comparison with EM data and conventional ANN model trained with training data of different sizes. The geometrical variables for three junctions are as follows.

- Geometry 1: $g = 19.5$ mm, $d_h = 1.8$ mm, $d_r = 2.7$ mm.
- Geometry 2: $g = 20.5$ mm, $d_h = 2.8$ mm, $d_r = 3.1$ mm.
- Geometry 3: $g = 22.5$ mm, $d_h = 3.0$ mm, $d_r = 3.7$ mm.

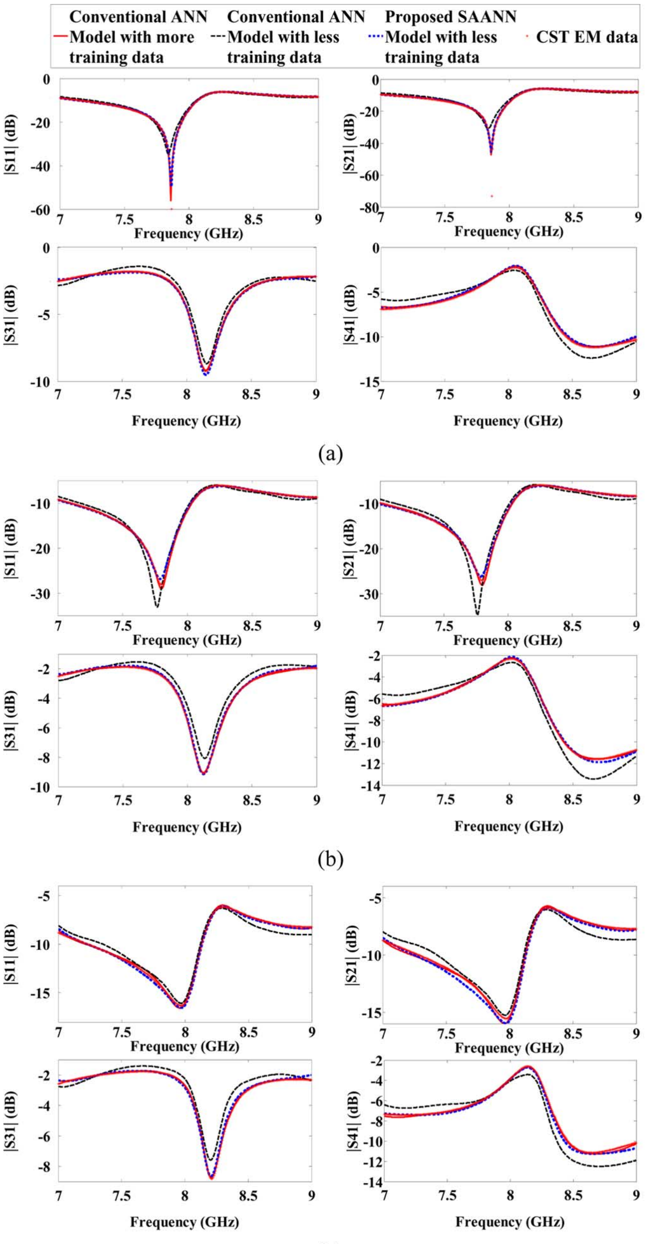
As shown in Fig. 18, broadband accuracy of the proposed SAANN parametric model is confirmed by its good agreement with EM data in terms of S_{11} S_{21}, S_{31}, and S_{41}, and even these geometries are never used in the training process.

Here, we show another benefit of this technique where the trained model can accurately predict the derivative information of the junction responses with respect to geometrical variables. As shown in Fig. 19, we provide the comparison of derivatives of the real part of S_{11} and S_{31} with respect to sensitivity variables g by the proposed SAANN parametric model and CST sensitivity analysis at geometries #1, #2, and #3, respectively. As shown in this figure, the proposed SAANN model can accurately predict the derivative information, which is close to those obtained from CST sensitivity analysis, even though such geometries are never used in the training process.

C. *Parametric Modeling of a Cavity Filter*

In this example, the proposed SAANN technique is applied to develop the parametric model of a family of microwave cavity filters, as shown in Fig. 20, where H_{c1}, H_{c2}, and H_{c3}, which represent the heights of the cylinders, respectively, are responsible for tuning the frequencies in the cavity, positioned at the cavity centers.

The structure of the proposed SAANN parametric model for the cavity filter example is shown in Fig. 21. The structure of the SAANN parametric model for microwave cavity filters is shown in Fig. 20. In this example, this SAANN parametric model has four inputs, i.e., $\boldsymbol{x} = [H_{c1},\ H_{c2},\ H_{c3}\ \omega]^T$, which include three geometrical variables H_{c1}, H_{c2}, andH_{c3} defined in Fig. 20 and frequency ω. In this example, all three input geometrical variables are all set as the sensitivity variables. This SAANN model combining the original and adjoint neural networks used for training has 20 outputs, i.e., $[RS_{11}\ IS_{11}\ RS_{12}\ IS_{12}\ \frac{dRS_{11}}{dH_{c1}}\ \frac{dRS_{11}}{dH_{c2}}\ \frac{dRS_{11}}{dH_{c3}}\ \frac{dRS_{11}}{d\omega}\ \frac{dIS_{11}}{dH_{c1}}, \ldots, \frac{dIS_{12}}{d\omega}]^T$, which are the derivatives of the real and imaginary parts of S_{11} and S_{12} to four inputs (including frequency). The sensitivity analysis in the CST EM simulator is performed to obtain the derivatives of real and imaginary parts of S_{11} and S_{12} to three sensitivity variables. Since the frequency ω is not sensitivity variables in CST EM simulation, the corresponding outputs from the SAANN parametric model are left as free variables in the model training process. This is achieved by setting the training weights for $[\frac{dRS_{11}}{d\omega}\ \frac{dIS_{11}}{d\omega}\ \frac{dRS_{12}}{d\omega}\ \frac{dIS_{12}}{d\omega}]^T$ as 0 in our training program. The frequency range is from 0.65 to 0.7 GHz with a step size of 1.5 MHz. The data range of training data and testing data is defined in Table V. Partial orthogonal design of the experiments method is used to determine the size of training and testing data.



(a)

(b)

(c)

Fig. 18. Comparison of the magnitude in decibels of S_{11} S_{21} S_{31} and S_{41} of the proposed SAANN model, CST EM data and conventional ANN model with less or more training data for three different geometries: (a) #1, (b) #2, and (c) #3 for the junction example. As shown in this figure, the proposed technique obtains more accurate model with less training data than conventional ANN technique. The match between proposed SAANN with original EM data is good even though the testing geometries used in the figures are never used in training.

Table VI shows final results of training in terms of the average training and testing error of the final trained model and its comparison with the conventional ANN model, which was trained without using EM derivative data. Two sets of training data were used in order to compare the effect of training with respect to different sizes of training data. One set of training data has 120

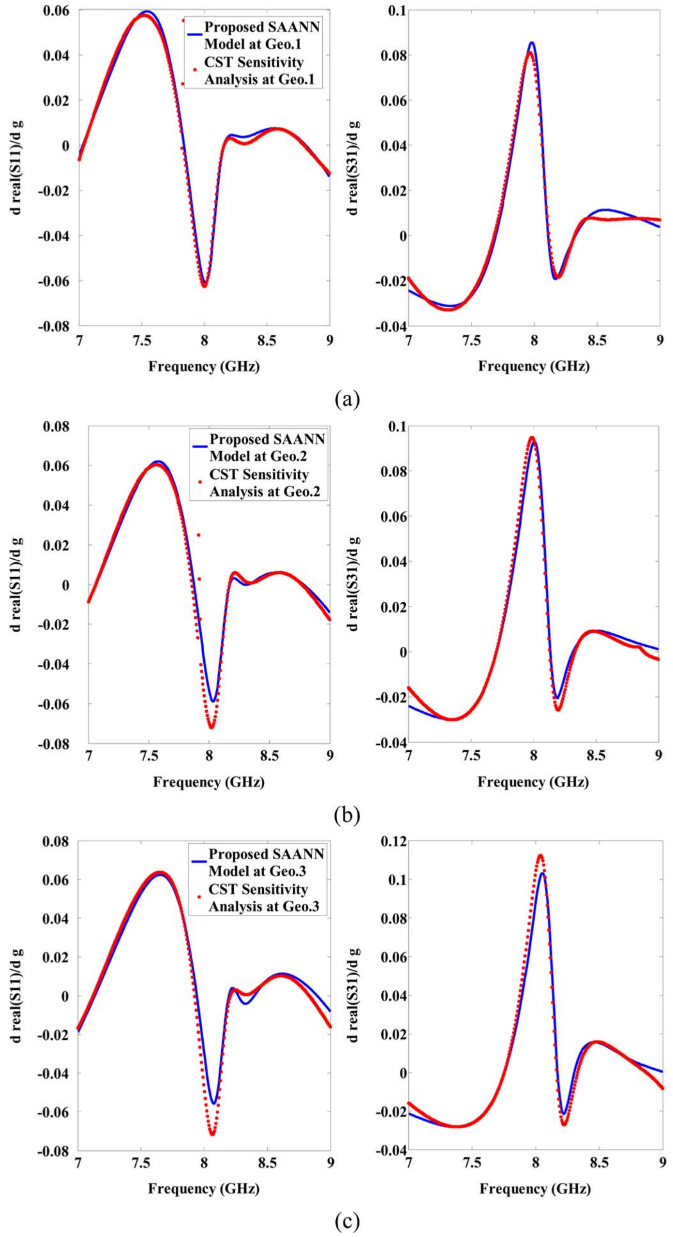


Fig. 19. Comparison of the derivative information of the real part of S_{11} and S_{31} to sensitivity variables $\boldsymbol{g}$ by the proposed SAANN model and CST sensitivity analysis for $dRS_{11}/\mathrm{d}g$ and $dRS_{31}/\mathrm{d}g$ at geometries: (a) #1, (b) #2, and (c) #3 for the junction example. As shown in this figure, the proposed SAANN model can accurately predict the derivative information, even though such a geometry is never used in the training process.

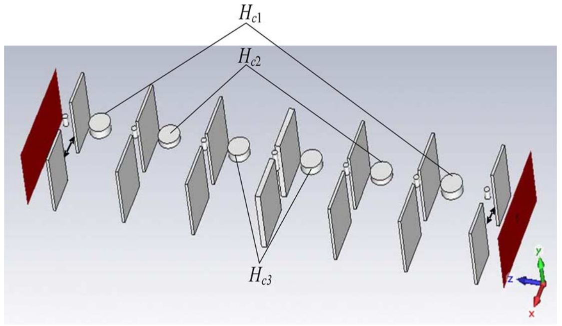


Fig. 20. Structure of a microwave cavity filter and geometrical parameters used for generating training data for parametric modeling example.

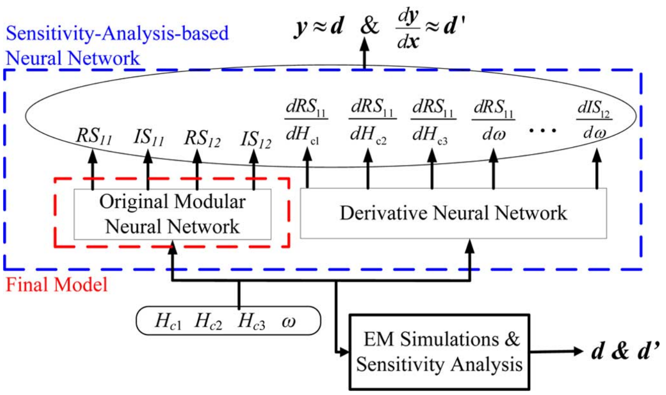

Sensitivity-Analysis-based Neural Network
$y \approx d$ & $\frac{dy}{dx} \approx d'$
RS_{11} IS_{11} RS_{12} IS_{12} $\frac{dRS_{11}}{dH_{c1}}$ $\frac{dRS_{11}}{dH_{c2}}$ $\frac{dRS_{11}}{dH_{c3}}$ $\frac{dRS_{11}}{d\omega}$ … $\frac{dIS_{12}}{d\omega}$
Original Modular Neural Network
Derivative Neural Network
Final Model
H_{c1} H_{c2} H_{c3} ω
EM Simulations & Sensitivity Analysis
d & d'

Fig. 21. Structure of the proposed SAANN parametric model for the cavity filter example.

TABLE V
DEFINITION OF TRAINING AND TESTING DATA FOR CAVITY FILTER EXAMPLE

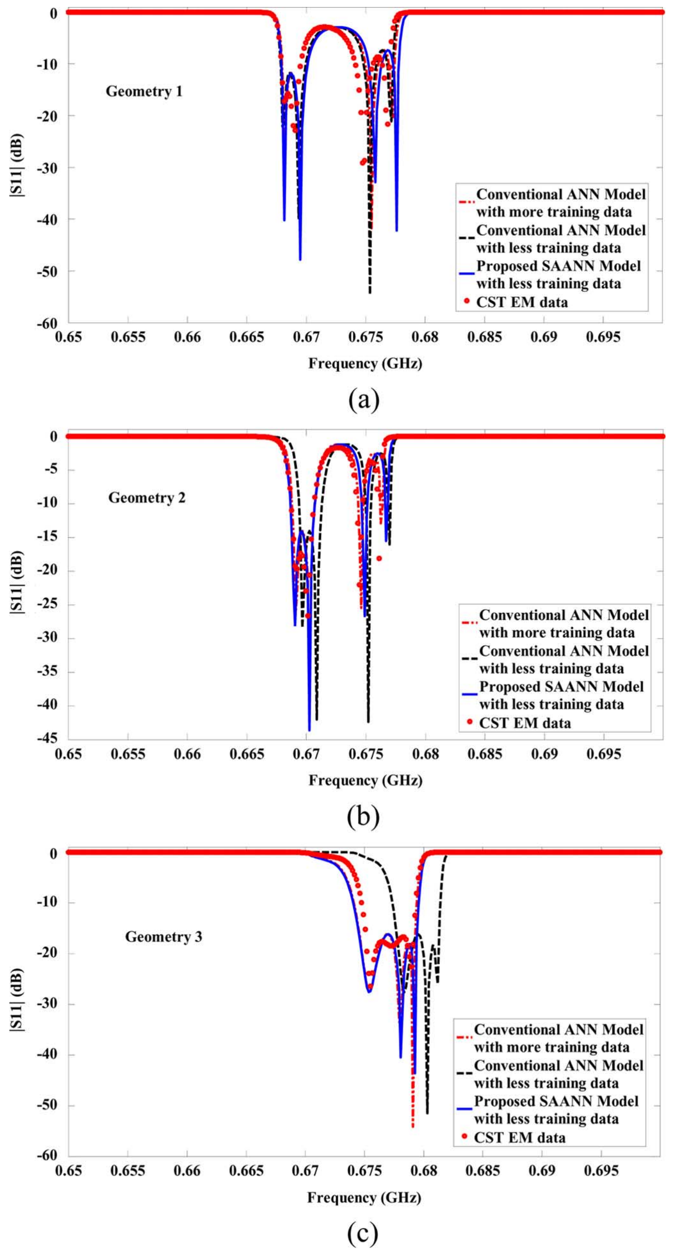
Parameters		Training data			Testing data		
		Min	Max	Step	Min	Max	Step
Sensitivity Variables	H_{c1} (mm)	25	29	1	25.5	28.5	1
	H_{c2} (mm)	16.5	20.5	1	17	20	1
	H_{c3} (mm)	23	27	1	23.5	26.5	1

TABLE VI
TRAINING AND TESTING RESULTS FOR CAVITY FILTER EXAMPLE

Model Type	Original Neural Network Structure	Average Training Error	Average Testing Error
Conventional ANN Model using 120 sets of training data	4-25-20-8	1.17%	1.71%
Conventional ANN Model using 50 sets of training data	4-25-20-8	1.15%	5.41%
Proposed SAANN Model using 50 sets of training data	4-25-20-8	1.47%	1.59%

samples (i.e., training with more training data), and another set has only 50 samples (i.e., training with less training data). From this table, we can see that with more training data, the conventional ANN model (i.e., trained without sensitivity information) can obtain a small training error and a small testing error. With less training data, the conventional ANN model trained without sensitivity information cannot obtain small testing error even though the training error is small. In contrast, the proposed SAANN model can obtain a small training error and a small testing error even with less training data. This is because the SAANN technique incorporates not only the input–output behavior of the modeling problem, but also the derivative information from sensitivity analysis into the model training process. Therefore, using sensitivity information, we can obtain an accurate model using less training data than without using sensitivity information.

The SAANN model development cost for this cavity filter example, including training data generation time (50 sets of training data) and model training time, is about 17.2 h and for the conventional ANN model development (120 sets of training

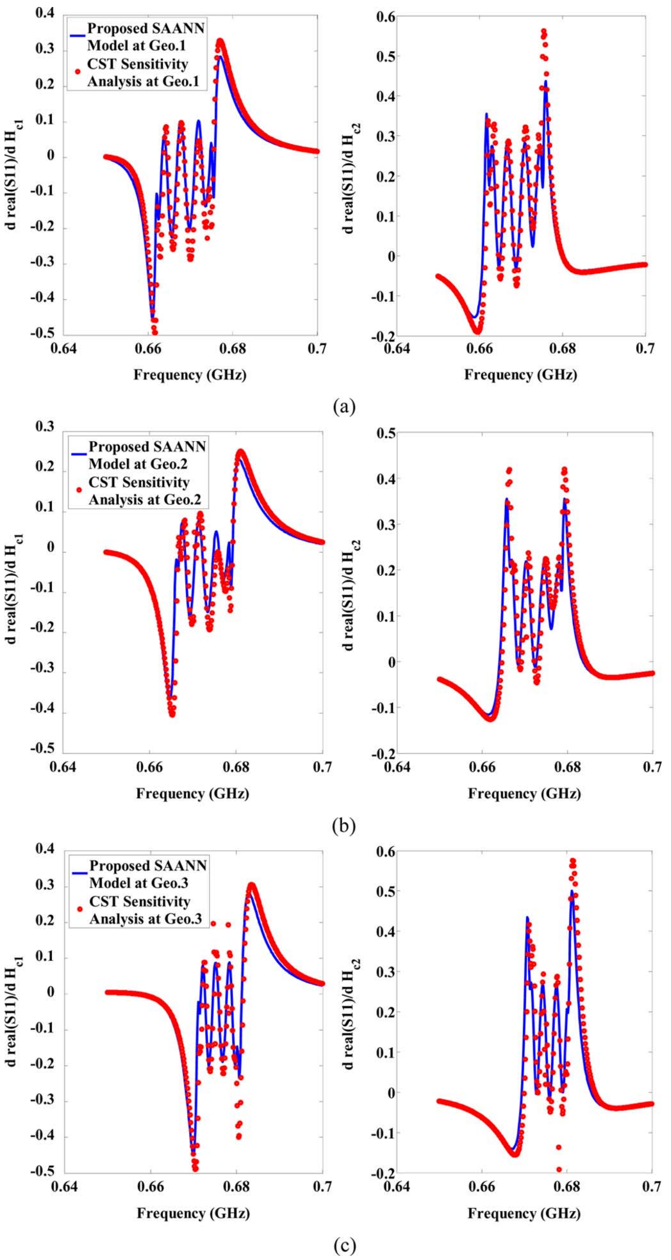


(a)

(b)

(c)

Fig. 22. Comparison of the magnitude in decibels of S_{11} of the proposed SAANN model, CST EM data, and conventional ANN model with less or more training data for three different geometries: (a) #1, (b) #2, and (c) #3 for the cavity filter example. As shown in this figure, the proposed technique obtains more accurate model with less training data than the conventional ANN technique. The match between the proposed SAANN with original EM data is good even though the testing geometries used in the figure are never used in training.



(a)

(b)

(c)

Fig. 23. Comparison of the derivative information of the real part of S_{11} to sensitivity variables H_{c1}, H_{c2} by the proposed SAANN parametric model and CST sensitivity analysis for $dRS_{11}/\mathrm{d}H_{c1}$, and $dRS_{11}/\mathrm{d}H_{c2}$ at geometries: (a) #1, (b) #2, and (c) #3 for the cavity filter example. As shown in this figure, the proposed SAANN model can accurately predict the derivative information, which are much closer to those obtained from CST sensitivity analysis, even though such geometry is never used in the training process.

geometries) is about 37.5 h. This further demonstrates that using the proposed technique, we speedup the model development time. Note that the training is a one-time investment, and the benefit of using the model accumulates when the model is used over and over again.

Fig. 22 depicts the outputs of the proposed SAANN parametric model for three different filter geometries #1, #2, and #3, and its comparison with CST EM data and conventional ANN model trained with training data of different sizes. The geometrical variables for three filters are as follows.

- Geometry 1: $H_{c1} = 27$ mm, $H_{c2} = 18.9$ mm, $H_{c3} = 25$ mm.
- Geometry 2: $H_{c1} = 28.8$ mm, $H_{c2} = 19.3$ mm, $H_{c3} = 25.6$ mm.
- Geometry 3: $H_{c1} = 27.8$ mm, $H_{c2} = 17.5$ mm, $H_{c3} = 24$ mm.

As shown in Fig. 22, broadband accuracy of the proposed SAANN parametric model is confirmed by its good agreement with EM data in terms of S_{11} even these geometries are never used in the training process.

Here, we further show that the trained model can accurately predict the derivative information with respect to geometrical variables. As shown in Fig. 23, we provide the comparison of the derivatives real part of S_{11} to sensitivity variables H_{c1}, H_{c2}, and H_{c3} by the proposed SAANN parametric model and CST sensitivity analysis at geometries #1, #2, and #3, respectively. As shown in this figure again, the proposed SAANN parametric model can accurately predict the derivative information, which is much closer to those obtained from CST sensitivity analysis, even though such geometries are never used in the training process.

V. Conclusion

In this paper, a novel SAANN technique for developing a parametric model of microwave passive components has been presented. Using sensitivity information, this technique introduces a new way to develop accurate neural-network models with less EM data than that without using sensitivity. The parametric SAANN models are well suited for purpose of establishing an EM component library, where the trained models can be reused again and again for microwave passive components design with different specifications. The SAANN technique can also provide sensitivity information with respect to geometric parameters, which are not sensitivity variables in the original EM simulator. Therefore, the method also helps to extend sensitivity analysis beyond the variable limits in EM simulators.

Acknowledgment

The authors thank Dr. P. Thoma and Dr. S. Reitzinger, both with Computer Simulation Technology (CST) AG, Darmstadt, Germany, for collaboration and technical input to this project, and for providing CST Microwave Studio for this research.

References

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

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

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

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

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

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

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

[8] S. Koziel and J. W. Bandler, "A space-mapping approach to microwave device modeling exploiting fuzzy systems," *IEEE Trans. Microw. Theory Techn.*, vol. 55, no. 12, pp. 2539–2547, Dec. 2007.

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

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

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

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

[13] CST Microwave Studio. CST AG, Darmstadt, Germany, 2010. [Online]. Available: http://www.cst.com

[14] HFSS. Ansoft Corporation, Canonsburg, PA, USA, 2007. [Online]. Available: http://www.ansoft.com/products/hf/hfss/

[15] N. K. Nikolova, J. Zhu, D. Li, M. H. Bakr, and J. W. Bandler, "Sensitivity analysis of network parameters with electromagnetic frequency-domain simulators," *IEEE Trans. Microw. Theory Techn.*, vol. 54, no. 2, pp. 670–681, Feb. 2006.

[16] Q. S. Cheng, J. W. Bandler, N. K. Nikolova, and S. Koziel, "Fast space mapping modeling with adjoint sensitivity," in *IEEE MTT-S Int. Microw. Symp. Dig.*, Baltimore, MD, USA, Jun. 2011.

[17] M. H. Bakr, N. K. Nikolova, and P. A. W. Basl, "Self-adjoint S-parameter sensitivities for lossless homogeneous TLM problems," *Int. J. Numer. Model.*, vol. 18, no. 6, pp. 441–455, Nov. 2005.

[18] O. S. Ahmed, M. H. Bakr, X. Li, and T. Nomura, "A time-domain adjoint variable method for materials with dispersive constitutive parameters," *IEEE Trans. Microw. Theory Techn.*, vol. 60, no. 10, pp. 2959–2971, Oct. 2012.

[19] N. Uchida, S. Nishiwaki, K. Izui, M. Yoshimura, T. Nomura, and K. Sato, "Simultaneous shape and topology optimization for the design of patch antennas," in *Proc. Antennas Propag.*, Mar. 2009, pp. 103–107.

[20] M. H. Bakr, M. Ghassemi, and N. Sangary, "Bandwidth enhancement of narrow band antennas exploiting adjoint-based geometry evolution," in *Proc. IEEE Int. Antennas Propag. Symp.*, Jul. 2011, pp. 2909–2911.

[21] A. Khalatpour, R. K. Amineh, Q. S. Cheng, M. H. Bakr, N. K. Nikolova, and J. W. Bandler, "Accelerating space mapping optimization with adjoint sensitivities," *IEEE Microw. Wireless Compon. Lett.*, vol. 21, no. 6, pp. 280–282, Jun. 2011.

[22] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, , D. E. Rumelhart and J. L. McClelland, Eds., *Learning Internal Representation by Error Propagation, Parallel Distributed Processing*. Cambridge, MA, USA: MIT Press, 1986, vol. I, pp. 318–362.

[23] O. Stan and E. Kamen, "A local linearized least squares algorithm for training feedforward neural networks," *IEEE Trans. Neural Netw.*, vol. 11, no. 2, pp. 487–495, Mar. 2000.

[24] Y. Xu, K.-W. Wong, and C.-S. Leung, "Generalized RLS approach to the training of neural networks," *IEEE Trans. Neural Netw.*, vol. 17, no. 1, pp. 19–34, Jan. 2006.

[25] K. C. Lee, "Application of neural network and its extension of derivative to scattering from a nonlinearly loaded antenna," *IEEE Trans. Antennas Propag.*, vol. 55, no. 3, pp. 990–993, Mar. 2007.

[26] Q. J. Zhang, "NeuroModeler plus," Dept. Electron., Carleton Univ., Ottawa, ON, Canada, 2005.

[27] S. R. Schmidt and R. G. Launsby, "Understanding industrial designed experiments," Air Force Acad., Colorado Springs, CO, USA, 1992.

Sayed Alireza Sadrossadat (S'12) received the B.S. degree in computer engineering from the College of Engineering, University of Tehran, Tehran, Iran, in 2007, the Masters degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2010, and is currently working toward the Ph.D. degree in electronics at Carleton University, Ottawa, ON, Canada.

His current research interests include computer arithmetic, probabilistic design, yield maximization, and neural network technology for circuits and systems design.

Yazi Cao (M'09) received the B.S. and Ph.D. degrees in electrical engineering from Wuhan University, Hubei, China, in 2004 and 2008, respectively.

Since 2009, he has been a Postdoctoral Fellow with the Department of Electronics, Carleton University, Ottawa, ON, Canada. His research interests include neural networks and the design and modeling of RF/microwave circuits and planar antennas.

Qi-Jun Zhang (S'84–M'87–SM'95–F'06) received the Ph.D. degree in electrical engineering from McMaster University, Hamilton, ON, Canada, in 1987.

In 1990, he joined the Department of Electronics, Carleton University, Ottawa, ON, Canada, where he is currently a Professor. He has authored or couathored over 200 publications. He is a member of the Editorial Board of the *International Journal of Numerical Modeling*. He is an Associate Editor for the *International Journal of RF and Microwave Computer-Aided Engineering*. His research interests are neural network and optimization methods for high-speed/high-frequency circuit design.

Dr. Zhang is a Fellow of the Electromagnetics Academy. He 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 (CAD) (MTT-1) of the IEEE Microwave Theory and Techniques Society (IEEE MTT-S).