Automated model generation for electronic devices using neural network approaches

Q J Zhang[1,3], W Na[2], M Li[1], Q Ding[1] and G Wu[1]

[1]China Communication Microelectronics Technology Co Ltd, Shenzhen, China
[2]Faculty of Information Technology, Beijing University of Technology, Beijing, China

[3]E-mail: qjzhang64@163.com

Abstract. This paper describes artificial neural network approaches to convert the conventionally expensive process of electronic modelling into an automated model generation process. Artificial neural networks are trained using machine learning algorithm to learn the electronic behaviour, and the trained neural network becomes a model to help predict the electronic device behaviour. The automated model generation algorithm performs adaptive data sampling to determine the amount and the distribution of training data needed to train neural networks. The algorithm also determines the number of hidden neurons needed to achieve a compact and accurate model. Also incorporated into the automated model generation method is an efficient interpolation approach to make the process much faster. The objective of the described method is to generate a compact neural-network based model with better accuracy and in less time than conventional approach. Examples of automated modeling of radio-frequency and microwave filters used in wireless electronic systems are described showing the advantage of this technique.

1. Introduction

Artificial neural networks are important vehicles in data processing and machine learning. In addition to popular applications such as image processing, speech recognition and control, artificial neural networks have also found interesting applications in engineering fields such as electronic modeling and design. Artificial neural networks are trained to learn the behavior of high-frequency electronic devices through machine learning algorithms. The trained neural networks can then represent the model of electronic devices [1,2]. Because neural networks can extract underline relationships from electronics data, the neural network is a much more efficient way to evaluate electronic design solutions than direct look-up table approach using measurement data. Various approaches to apply neural networks to electronic modeling and data processing have been presented in the past, such as radio-frequency and microwave modeling and design [1,2], high-power high-frequency transistor modeling [3], electromagnetic modeling/optimization [4,5] and waveguide filter inverse modeling [6]. The combination of electronic knowledge and neural networks led to knowledge-based neural networks for electronic modeling [7] and space mapping concepts [8] has been applied to map empirical models to accurate device behavior through neural networks. Another important direction is automation of neural network model development which was introduced to integrate all subtasks like data generation, neural-network selection, training and test into one unified framework [9]. In addition to making neural network training more efficient, the automated model generation also addresses two critical problems of neural network training, i.e., determining the suitable amount of training data and the suitable number of hidden neurons. The neural network training is performed stage by stage, where

in each stage the amount of training data or the neural network size is adjusted, and neural network is re-trained. In this way, the suitable amount of training data and the suitable number of hidden neurons are automatically determined as part of the training process. However, process in [9] involves repetitive training of neural networks while detecting if the training data is sufficient in different regions of the nonlinear electronic parameter space. This reduces the model development efficiency.

This paper describes an advance on automated model generation [10]. We develop systematic algorithm to incorporate data sampling, data generation, neural network training and testing in a more efficient manner. We also incorporate interpolation techniques to avoid training the intermediate neural networks during the automated model generation process. Training is performed only after data sampling process has finished, making the automated modeling generation process faster than the previously published approaches. In addition, we add a procedure to reduce the number of hidden neurons so that the guess of initial neural-network structure can be more flexible and the final size of the neural-network model is more compact. Demonstration of automated model generation for a radio-frequency and microwave filters are also described.

2. Methodology

2.1. Training data and data generation for high-frequency electronics modeling

Let $\boldsymbol{x}$ represent a n-vector containing physical/geometrical parameters of an electronic device, e.g., gate length and gate width of a high-frequency transistor, or width and spacing of a transmission line. Let $\boldsymbol{y}$ represent a m-vector containing the responses of the device/circuit under consideration, e.g., drain current of a transistor, or scattering parameters of a transmission line. Let the physics/ electromagnetics relationship between $\boldsymbol{x}$ and $\boldsymbol{y}$ be represented as $\boldsymbol{y} = \boldsymbol{y}(\boldsymbol{x})$. This relation can be highly nonlinear and multi-dimensional. The theoretical model for this relationship may not be available (e.g., a new semiconductor device), or theory may be too complicated to implement, or the theoretical model may be computationally too intensive for online microwave design and repetitive optimization (e.g., 3D full-wave electromagnetic analysis inside a Monte Carlo statistical design loop).

To overcome these problems, we aim to develop a fast and accurate artificial neural network model for the relation $\boldsymbol{f}$ by teaching/training the artificial neural network to learn the electronic behaviour from a set of data. Let the training data be denoted as input-output sample pairs, $\{(\boldsymbol{x}^{(i)}, \boldsymbol{d}^{(i)}), i=1,2, \ldots, N_d\}$, where $\boldsymbol{d}^{(i)}$ represents the data for output $\boldsymbol{y}$ when the inputs are $\boldsymbol{x}^{(i)}$, and N_d represents the number of training data. The measured/simulated data of the outputs and the inputs are related by,

$$\boldsymbol{d}^{(i)} = \boldsymbol{f}(\boldsymbol{x}^{(i)}) \tag{1}$$

The data generation can be done either by measurement of the device, or by simulation using electromagnetic simulators. Typically, data for nonlinear active devices such as Gallium Nitride transistors are obtained by measurement. The scheme to gather measurement data is that we first apply different voltages to the transistor gate and drain, and then measure the drain current to obtain current voltage data. Secondly we generate small-signal data by measuring the scattering parameter data (called the S-parameter data) over a frequency range, and re-perform such measurement again and again for different operating bias voltages. Thirdly, we perform large-signal measurement by exciting the device with large signal at the gate, and measure the distorted large-signal at the drain. In this way, we obtain the measurement data in the form of harmonic data including signal at fundamental frequency, second-harmonic frequency, and all the high-order harmonic frequencies. This harmonic data measurement is repeated for different operating bias points, different values of fundamental frequencies and different levels of input signal power. Consequently, a large amount of data is collected, which contain valuable information of the device internal dynamic relationships, for which no explicit formula is available.

Data generation for passive electronic components are typically achieved through simulation. This data can be supplemented by measurement data. The reason is that the typical design variables for passive components are geometrical variables such as height of an iris window in a waveguide resonant cavity, or length and width of a microstrip in a filter. It is expensive to manufacture a large

number of passive components corresponding to different geometries just for purpose of data collection. Therefore, electromagnetic simulation become typical choice for collecting data for passive high-frequency electronic components. The amount of data becomes large if more geometrical variables (e.g., length 1, length 2, …, width 1, width 2, … for multiple coupled microstrip lines) are needed. In this case, we need to obtain data for various combinations of values from many geometrical parameters.

2.2. Training of neural networks with automated model generation algorithm

To efficiently use data to train the neural networks, our automated model generation process includes initial data generation, neural network training, neural network size adjustment, additional data generation, and neural network testing. Incremental data generation is performed whenever and wherever data are needed as decided by the model generation algorithm during neural network training process.

The overall automated model generation process proceeds in stage-wise process [10]. Each stage includes data generation, neural-network structure adaptation, training and test. Different stages may have different data and different neural-network size. Our process uses three sets of data, namely training, interpolation and test data. Training data are used to adjust neural-network weights during training. Interpolation data are used to create local models in subregions of the model input space R^0 to assess the adequacy of training data during training process. Additional training data are added upon detecting data inadequacy by the interpolation criteria. Test data are used to check the quality of the final neural model.

Let $(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})$ represent the ith data. Let L^k, I^k, and T^k denote training, interpolation and test data respectively in the kth stage. Let H^k denote the number of hidden neurons of neural network S^k in the kth stage. The algorithm regards the original input space R^0 as one region at first. If the test error in the region is large, the region would be divided subregions and new data are generated. In each stage, the algorithm compares the test error in each subregion. The subregion with the maximum test error is chosen as the worst region R^*. Then R^* would be further divided into 2^n new subregions in the next stage. In order to avoid training intermediate neural networks in each stage, we evaluate the test error of various subregions using interpolation approaches during each stage of the automated model generation process. Different subregions have different interpolations. When calculating the interpolation in one region, the region is called the interpolation region. Even-though the electronic behavior over the entire region of design space is complex and nonlinear therefore requiring a neural networks to learn, the interpolation space is usually in a small local subregion. Therefore, a simpler interpolation function $g(\boldsymbol{x})$ is sufficient for interpolation space. In this way we speedup the overall model generation process. In our work, we use polynomial interpolation. Let $g(\boldsymbol{x})$ be the polynomial interpolation function with N terms. Suppose $\boldsymbol{p}$ is a vector of coefficients in the interpolation function, and $\boldsymbol{h}(\boldsymbol{x})$ is a vector of polynomial terms in the multi-dimensional space of $\boldsymbol{x}$. For example, for a second order polynomial in two variables, $n = 2$, $N = 6$, $\boldsymbol{p} = [p_1, p_2, \_, p_6]$ and $\boldsymbol{h} = [1, x_1, x_2, x_1x_2, x_1, x_2]$.

In general for neural networks with multiple outputs, say m outputs, $\boldsymbol{p}$ would be an N x m matrix. To compute the test error for an interpolation region, we select M training data in and around the interpolation region to form matrices **A** and **B**, such that

$$\mathbf{A}\boldsymbol{p} = \mathbf{B}. \tag{2}$$

where **A** has N columns and M rows, and **B** has m columns and M rows, where N, M and m are the number of elements in $\boldsymbol{h}$, the number of training data selected, and the number of model outputs, respectively. The elements in **B** are output values $\boldsymbol{y}^{(i)}$ from M training data. To determine **A**, the locations of these M training points $\boldsymbol{x}^{(i)}$ are related to $\boldsymbol{x}^*$, which is at the center of the worst region R^*. $\boldsymbol{x}^{(i)}$ is obtained as

$$\boldsymbol{x}^{(i)} = \boldsymbol{x}^* + \boldsymbol{Q}_i \, (\boldsymbol{x}_{max} - \boldsymbol{x}_{min})/2^r \tag{3}$$

where r is the number of times the original region was divided to reach the present worst subregion R^*, $\boldsymbol{x}_{\min}$ and $\boldsymbol{x}_{\max}$ are minimum and maximum values of the variables in the original $\boldsymbol{x}$ space, $\boldsymbol{x}^*$ is the test

point in R^*, and $\boldsymbol{Q}_i$ is a $n \cdot n$ diagonal matrix with the values 0, +1, or -1 on its diagonal. Different combinations of 0, +1, and -1 in the diagonal of $\boldsymbol{Q}_i$ allow us to obtain different training points. After normalizing the $\boldsymbol{x}$ and $\boldsymbol{y}$ values, the **A** matrix can be derived as a constant matrix, containing 0, +1, and -1. Additional scaling factors can be applied to particular rows in $\boldsymbol{A}$ in order to emphasis the importance of fitting particular data points. The least squares solution of normalized version of (2) is

$$\boldsymbol{p} = (\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{B} \tag{4}$$

Subsequently the interpolation function is obtained for the subregion R^*. Then, the test error E_{In} of the interpolation regions is calculated by comparing the $g(\boldsymbol{x})$ and test data. The subregion with the largest test error will be designated as the new worst region R^*. Automated model generation process will then enter the next stage where the new worst region is further divided, forming a new set of subregions, and the process described in (2)-(4) is repeated. In this way, we take advantage of availability of training data to produce localized interpolation function, avoiding the training of neural network during intermediate stages of automated model generation process, and speeding up model development. Neural network training is performed only during the last few stages of automated model generation process. The final model is a neural network valid in the entire region. We also add a procedure reducing the number of hidden neurons after good-learning is detected, as shown in figure 1. In the figure, E^{d} represents user-defined error threshold for training the model. S^k denotes the neural network in the kth stage of automated model generation process, and H^k represents the number of hidden neurons of S^k. Symbols E_{In} and E_l denote interpolation error w.r.t. test data, and training error w.r.t. training data, respectively.

The algorithm will recursively reduce the number of neurons until the error of the reduced neural network starts to increase. This procedure will allow the guess of the initial number of hidden neurons to be more flexible, and make the final neural model more compact. The overall algorithm can start from an initial set of data and initial neural network, and ultimately find the most compact neural network to learn the given electronic problem using data collected at the most critical regions of the input parameter space.

3. Experimental / Theoretical Results and Discussions

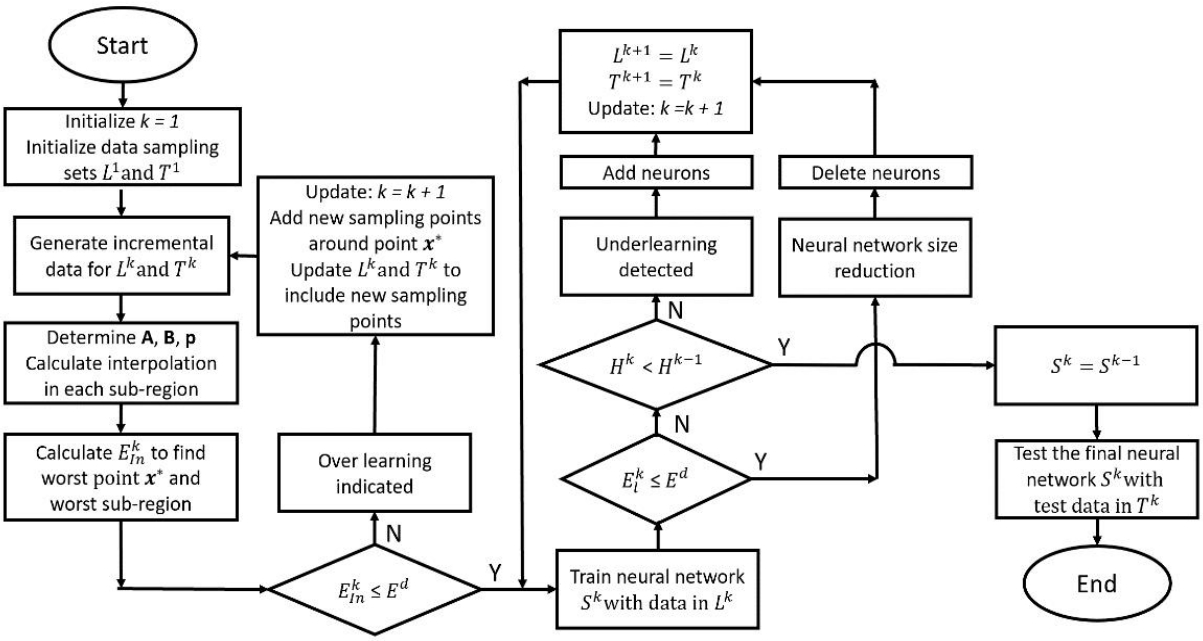

Figure 1. Flow chart of the proposed automated model generation algorithm, including data generation, neural network training, neural network size adjustment and neural network testing.

In this section, we describe results of applying the automated model generation method to an example of neural network modeling of a bandpass filter in wireless communication systems [10]. More specifically, the application of the method is demonstrated by the development of a parametric model of a High-Temperature Superconductor (HTS) quarter-wave parallel coupled-line microstrip filter [10,11] used in radio-frequency and microwave wireless electronic systems [8]. The HTS filter structure is illustrated in figure 2. In order to generate training data for neural networks to learn, the filter geometrical formulation is setup in a 3-dimension electromagnetic simulator. The electromagnetic simulator is executed upon demand to generate training data at the worst sub-regions of the modeling space in each stage as determined by the automated model generation algorithm.

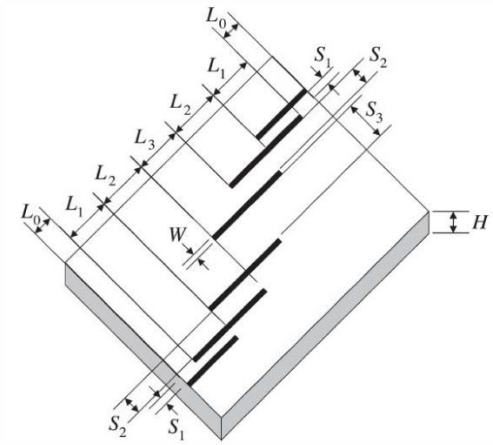

Figure 2. The structure of the HTS coupled line microstrip bandpass filter example.

The response behavior of the radio-frequency/microwave filter is best represented by the transfer function and the return loss of the filter. Since this filter operates in 4 GHz frequency range, conventional resistor/capacitor/inductor representation is not applicable. Electromagnetic oriented models are required to capture the subtle relationship between geometrical parameters and the filter responses. However, standard electromagnetic simulation is time-consuming costing the designer's expensive online design time. We use neural networks to learn from electromagnetic data of the filter. Using the automated model generation process, neural network models can be developed offline, including automated data generation offline and automated neural network training offline. The trained neural network is a fast and accurate model for designer's online use, providing fast estimation of electromagnetic behavior of the filter during filter design. In this way, filter design can be substantially faster.

The input parameters for the neural network model are $\boldsymbol{x} = [L_1, L_2, L_3, S_1, S_2, S_3]^T$, where L_1, L_2 and L_3 are the lengths of the parallel coupled-line sections and S_1 , S_2 and S_3 are the gaps between the sections. The outputs for the neural network are $\boldsymbol{y}=[S_{11}, S_{21}]^T$ where S_{11} represent the return loss of the filter, and S_{21} represent the transfer function of the filter in terms of the input-output wave signals.

Training data generation is done by an electromagnetic simulator, called CST Microwave Studio. In this example, there are six geometrical variables, i.e., $n = 6$. Matrix **A** has 70 columns meaning that the interpolation function has 70 coefficients or 70 polynomial terms (i.e., N=70). The number of rows of matrix A is 729, meaning that the number of training samples around the neighborhood of an interpolation space is 729 (i.e., M=729). Matrix **B** has 2 column and 729 rows. The worst region R^* is divided into 64 subregions, so there are 64 **A** matrices used during interpolation process. The automated modeling methods used 1394 electromagnetic training geometries. After 6 stages of the process in Figure 1, the final neural network training is finished. The test error of the final neural network is 1.59% representing a good accuracy for engineering use. Figure 3 shows the neural network response compared with original electromagnetic response, confirming that the neural network has good accuracy for predicting the electromagnetic behavior of the filter. Notice that the

neural network responses in figure 3 are calculated with two sets of testing geometry which are never used in training.

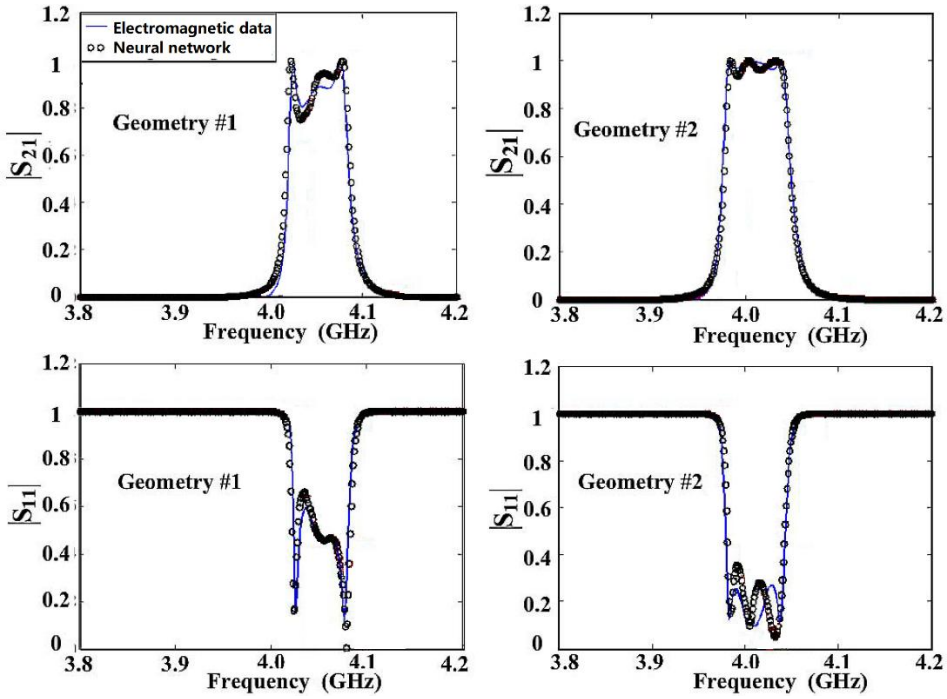

Figure 3. The comparison of neural network result versus original electromagnetic test data of the HTS bandpass filter at two different sets of geometrical values. For geometry #1, $\boldsymbol{x}$=[4.45, 4.81, 4.60, 0.52, 2.32, 2.0]^T (mm). For geometry #2, $\boldsymbol{x}$=[4.58, 4.96, 4.46, 0.58, 2.10, 2.11]^T (mm).

For comparison purpose, we also attempted conventional manual data generation and training of neural networks. To achieve similar test accuracy (1.59%), we endup with much more training data (4096 electromagnetic training geometries), and more training time (85 hours including data generation and neural network training). Using the automated model generation algorithm, the training data is judicially selected such that highly nonlinear sub-regions of the modeling space have more training samples, and smooth subregions have less training samples. In this way, only 1394 electromagnetic training geometries are sufficient to achieve same accuracy as manual training with 4096 training geometries. Subsequently the automated model generation process is much faster than manual data generation and training.

Further comparison is performed by comparing the method in this paper with the automated training algorithm of [9]. The method described in this paper achieved same accuracy of trained neural network as that of [9]. However, the model trained by the method in this paper used less stages (6 stages by the method in this paper versus 9 stages by the existing automated model generation of [9]). The total amount of time for developing the model including electromagnetic data generation and neural network training by the method in this paper is less than the existing automated model generation of [9] (22 hours by automated algorithm in this paper versus 26 hours by previous automated algorithm), and produced a more compact neural network. Other aspects for the comparison are equal between the automated model generation algorithm of this paper and the automated model generation algorithm of [9], including same optimization method for training, same electromagnetic simulator, the same geometrical configurations for the bandpass filters. In training the neural networks, the quasi-Newton algorithm, which is a second-order gradient based optimization algorithm, was used for training. The models trained by the method in this paper and that in [9] are both validated by comparing with original electromagnetic data simulated at 10001 frequency points across the entire frequency range of interests. The reason for a net speedup of model development by the method

described in this paper is due to the use of interpolation techniques in two stages of the automated model generation, bypassing intermediate training of neural networks and thus shortening the overall model development time. In this way, the model generation method described in this paper is faster than the previously published automation methods [9].

4. Conclusion

An artificial neural network approach has been described to learn and predict the electromagnetic behaviour of electronic components. The neural network training has been described through an automated model generation process which includes data sampling to determine the amount and the distribution of training data needed to train the neural network, and to determine the number of hidden neurons to obtain a compact and accurate model. We have also described an interpolation approach to improve the speed of the model development process. An example has been described for neural network modeling of radio-frequency and microwave filters to illustrate that the training can be carried out efficiently and the trained neural network model achieves good accuracy in predicting the electromagnetic behaviour of the filter. The potential impact of this work is to employ artificial neural networks to allow accurate electronic models to be developed in shorter time and by automation, subsequently speeding up the design cycle of electronic circuits. This work is useful for electronic designers in their electronic data processing, fast behaviour modelling and design of electronic components and systems.

References

[1] Rayas-Sanchez J E 2004 EM-based optimization of microwave circuits using artificial neural networks: the state-of-the-art *IEEE Trans. Microwave Theory Tech.* **52** p 420-35

[2] Kabir H, Zhang L, Yu M, Aaen P H, Wood J and Zhang Q J 2010 Smart modeling of microwave devices *IEEE Microwave Mag.* **11** p 105-18

[3] Prasad A, Thorsell M, Zirath H and Fager C 2018 Accurate modeling of GaN HEMT RF behavior using an effective trapping potential *IEEE Trans. Microwave Theory Tech.* **66** p 845-57

[4] Na W C, Liu W Y, Zhu L, Feng F, Ma J G and Zhang Q 2018 Advanced extrapolation technique for neural-based microwave modeling and design *IEEE Trans. Microwave Theory Tech.* **66** p 4397-418

[5] Feng F, Gongal-Reddy V, Zhang C, Ma J G and Zhang Q J 2017 Parametric modelling of microwave components using adjoint neural networks and pole-residue transfer functions with EM sensitivity analysis *IEEE Trans. Microwave Theory Tech.* **65** p 1955-75

[6] Zhang C, Jin J, Na W C, Zhang Q J and Yu M 2018 Multivalued neural network inverse modeling and applications to microwave filters *IEEE Trans. Microwave Theory Tech.* **66** p 3781-97

[7] Na W, Zhang Q J 2015 Unified automated knowledge-based neural network modeling for microwave devices *IEEE MTT-S Int. Conf. Numerical Electromagnetic and Multiphysics Modeling and Optimization* (Ottawa, Canada) August 2015

[8] Bandler J W, Cheng Q S, Dakroury S A, Mohamed A S, Bakr M H, Madsen Kaj and Sondergaard Jacob 2004 Space mapping: the state of the art *IEEE Trans. Microwave Theory Tech.* **5** p 337-61

[9] Devabhaktuni V K, Yagoub M C E and Zhang Q J 2001 A robust algorithm for automatic development of neural-network models for microwave applications *IEEE Trans. Microwave Theory Tech.* **49** p 2282-91

[10] Na W C, Zhang Q J 2013 Automated parametric modeling of microwave components using combined neural network and interpolation techniques *IEEE MTT-S Int. Microwave Symp.* (Seattle, Washington) June 2013

[11] Na W C, Zhang Q J 2014 Automated knowledge-based neural network modeling for microwave applications *IEEE Microwave and Wireless Components Letters* **24** p 499-501