www.czasopisma.pan.pl PAN www.journals.pan.pl

jet INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2022, VOL. 68, NO. 4, PP. 793–798
Manuscript received December 21, 2021; revised November, 2022. DOI: 10.24425/ijet.2022.143887

Power-Ground Plane Impedance Modeling Using Deep Neural Networks and an Adaptive Sampling Process

Chan Hong Goay, Zheng Quan Cheong, Chen Eng Low, Nur Syazreen Ahmad, and Patrick Goh

***Abstract*—This paper proposes a deep neural network (DNN) based method for the purpose of power-ground plane impedance modeling. A composite DNN model, which is a combination of two DNNs is used to predict the Z-parameters of power ground planes from their design parameters. The first DNN predicts the normalized Z-parameters whereas the second DNN predicts the original maximum and minimum values of the non-normalized Z-parameters. This allows the method to retain a high accuracy when predicting responses that have large variations across designs, as is the case with the Z-parameters of the power-ground planes. We use the adaptive sampling algorithm to generate the training and validation samples for the DNNs. The adaptive sampling algorithm starts with only a few samples, then slowly generates more samples in the non-linear regions within the design parameters space. The level of non-linearity of the regions is determined by a surrogate model which is also trained using the generated samples as well. If the surrogate model has poor prediction accuracy in a region, then the adaptive sampling algorithm will generate more samples in that region. A shallow neural network is used as the surrogate model for non-linearity determination of the regions since it is faster to train and update. Once all the samples have been generated, they will be used to train and validate the composite DNN models. Finally, we present two examples, a square-shaped power ground plane and a square-shaped power ground plane with a hollow square at the center to demonstrate the robustness of the DNN composite models.**

***Keywords*—adaptive sampling, deep neural networks, deep learning, power-ground plane, Z-parameters**

I. Introduction

The analysis of power distribution and quality on the power-ground planes in modern electronic circuits is one of the major challenges faced by circuit designers of high-speed systems. To ensure proper power integrity, the impedance parameters of the planes must be analyzed by performing simulations of the planes, in order to satisfy the maximum allowable impedance across the design frequencies [1]. Traditionally, this simulation is performed using a full wave simulator, which has a high accuracy. However, the computations necessary in a full wave simulator is computationally very expensive especially for highly complicated designs. Other than that, since the design process often has a lot of design parameters, multiple full-wave simulations will have to be performed, one for each set of unique design parameters. The number of simulations can get prohibitively big when the number of design parameters is large, thus making the whole design a very time consuming process.

Artificial neural networks (ANNs) have been applied to the field of RF and microwave circuit modeling for more than a decade, including for example the works in [2]–[5]. The main idea behind these works is to use the ANNs to replace the expensive full-wave simulators once they have being properly trained. Since the ANNs take almost no time during the prediction process, the cost of using ANNs mainly comes from generating the training and validation samples and the training process itself. The generation of training and validation samples can be very time consuming. Therefore, it is important to use a proper sampling technique to avoid the occurrences of oversampling and undersampling. Oversampling can increase the cost to construct the ANN model due to having redundancy in the samples and undersampling can result in a poor performing ANN model. The adaptive sampling algorithm is a sampling method that starts with a small number of samples within a design parameters space, and trains an ANN model with those samples. Then, it iteratively generates more samples in the regions with the worst prediction errors. The ANN model will also be updated iteratively with the new samples until its validation performance reaches the goal [6]. A modification to the adaptive sampling algorithm is proposed in [7], such that the algorithm is more balanced in terms of exploration and exploitation, and avoid getting ”stuck” in a region when the simulator produces erroneous results.

In the past, the most popular neural network structure for circuit modeling application is the shallow neural network which usually has only one or two hidden layers due to its simplicity and good performance. Recently, deep neural networks (DNNs) have gained popularity in the circuit modeling community as well. For example, DNNs have been used for the modeling of high-speed channels [8], [9], microwave filters [10], [11], and high power amplifiers [12]. Some studies show that the capability of shallow neural networks is very limited when compared to the DNNs, especially for modeling non-uniform highly complex functions or when there are missing

This work was supported by the Ministry of Higher Education, Malaysia, through the Fundamental Research Grant Scheme (FRGS) under Grant FRGS/1/2020/TK0/USM/02/7.

Chan Hong Goay, Nur Syazreen Ahmad, and Patrick Goh are with the School of Electrical and Electronic Engineering, Universiti Sains Malaysia, 14300 Nibong Tebal, Pulau Pinang, Malaysia. (Corresponding author: Patrick Goh, eepatrick@usm.my).

Zheng Quan Cheong is with the School of Electrical Engineering, Universiti Teknologi Malaysia, 81310 Skudai, Johor, Malaysia.

Chen Eng Low is with Aemulus Corporation, 11900 Bayan Lepas, Pulau Pinang, Malaysia.

values in the dataset [13], [14]. The readers are referred to [15] for the background and basics of DNNs.

In this paper, we present a method that uses the modified adaptive sampling algorithm for generating training and validation datasets, and then use the datasets to build the DNN models to model the Z-parameters of power-ground planes. We will demonstrate the capability of the DNN models using a two-dimensional (square-shaped power ground plane) and a three-dimensional (square-shaped power ground plane with a hollow square at the center) example.

II. Modified Adaptive Sampling

During initialization, the adaptive sampling algorithm defines the entire d-dimensional design parameters space as a single region, R, and training and validation samples are generated in the region. In this paper, we use a 2k-DoE distribution plus a center point for the training samples, and a star distribution for the testing samples. Then, these samples are used to construct an intermediate surrogate model which is used to determine the linearity of every region. The region with the largest validation error will be identified as the worst performing region and it will be split into 2^d regions of equal volume. Then, samples are generated in the new regions and appended to the existing samples. After that, the surrogate model will be updated using the newly generated samples and the algorithm will search for the new worst performing region again. This process is repeated until the stopping criteria is met. Figure 1 shows an example for a 2-dimensional design parameters space ($d = 2$), where the worst performing region is split into 4 regions with equal volume during the first to third iterations.

The original adaptive sampling only uses the validation errors to determine the performances of every region [6]. The modified adaptive sampling algorithm modifies this by assigning a value, n for each region where the n_i for the ith region, R_i is defined as:

$$n_i = \log_{2^d}\left(\frac{\text{volume of entire design parameters space}}{\text{volume of region } R_i}\right)+1. \tag{1}$$

The region error, E_i of the ith region is then defined as:

$$E_i = \frac{\text{validation error of } R_i}{n_i}. \tag{2}$$

The modified adaptive sampling algorithm will then select the region with the largest region error as the worst performing region. The purpose of dividing the validation errors with n is to prevent the algorithm from being "stuck" in a small outlier region. It can be seen from (1) and (2), when a region, R_i has a very small volume, its n_i value will be large, thus decreasing the chance of it being identified as the worst performing region. This can occur when the simulator gives erroneous results or when the data in that region behaves completely different than the rest of the region. The readers are referred to [7] for more details on the modified adaptive sampling algorithm. In this paper, we use a shallow neural network with only a hidden layer as our surrogate model because it has a faster training time than the DNNs. This is also based on the assumption that shallow neural networks have comparable performance to DNNs when the dataset is small [16]. We will discard the intermediate surrogate model, and build a DNN model once the sampling process is complete.

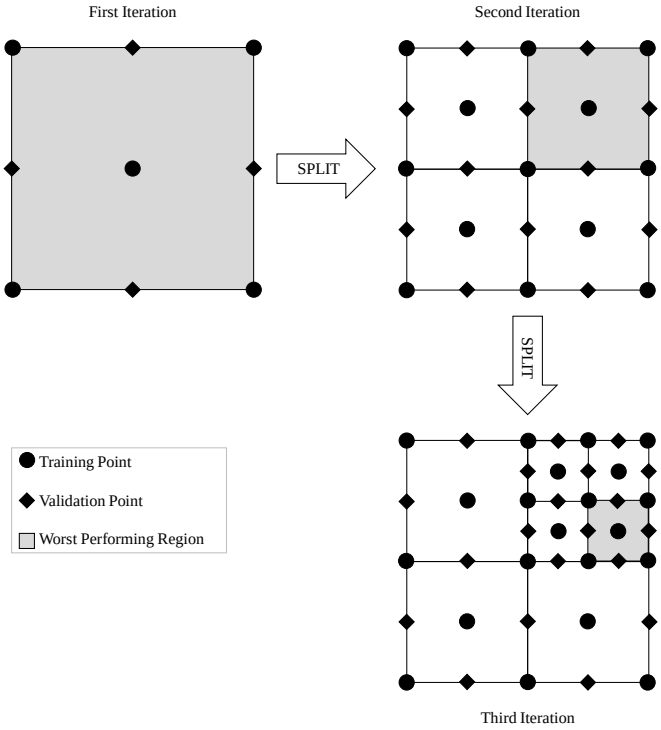

Fig. 1. The positions of training and validation samples from the first to third iterations of a 2-dimensional design parameters space

III. Power Ground Plane Modeling Using DNNs

A power-ground plane is a large layer of copper foil which distributes the direct current (DC) power to active devices and acts as a voltage reference on a multilayer printed circuit board (PCB) [17]. Power-ground planes are also referred to as power distribution networks (PDNs) in PCB designs, as they provide a current path for power transmission from the source to the loads on the PCB. A PDN should be designed to provide a low-noise power supply to the loads on the PCB within some peak voltage ripple to meet the system requirements of the IC [18], [19]. Although PDNs can be modeled as passive devices such as resistors, capacitors and inductors, or by transmission lines between source and loads [20], a more accurate result can be obtained by simulating the behavior of the PDNs directly from the physical dimensions and parameters. Figure 2 shows the structures of a square-shaped power ground plane and a square-shaped power ground plane with a hollow square at the center. The square-shaped power ground plane is represented by two design parameters: side length, l, and substrate height, h. On the other hand, the square-shaped power ground plane with a hollow square at the center is represented by three design parameters: l, h, and the ratio, r, between the side length of the outer square and the inner square.

In this paper, we use DNNs to model the Z-parameters of the power ground planes from the design parameters. However,

the Z-parameters of every set of design parameters can have vastly different magnitudes. For example, Figure 3 shows the magnitude of Z_{12} of four different power-ground planes. During the training process, the larger magnitude will have a larger effect on the training errors, which the network relies on for its weights update. In order to ensure that every design has the same level of importance during the training process, we will normalize the Z-parameters of each design, independently, to a maximum value of one and a minimum value of zero, as can be seen in Figure 4. However, normalization is not performed across the designs. As can be seen in Figure 5, the problem of the difference in magnitudes will still exist if normalization is performed across the designs.

Since the Z-parameters of every design are normalized independently, the information about the original magnitude of the Z-parameters is lost. Thus, we need another model to predict the original maximum and minimum values of the Z-parameters for every design. We propose the use of two DNN models for the power-ground plane modeling task, where model 1 predicts the normalized Z-parameters from the design parameters, x, and frequency, f, while model 2 predicts the maximum and minimum values for the Z-parameters from x, which will be used to perform the denormalization process. Figure 6 shows the structure of the proposed DNN-based composite model for the power ground plane modeling task, which consists of the DNN models and a denormalization block. All the DNN models are constructed using TensorFlow 2 [21]. We use the Adam optimizer [22], which is an adaptive learning rate optimization algorithm for training our DNN models. The Adam optimizer has low memory requirements, and is robust and well-suited to a wide range of non-convex optimization problems. Since the choice of hyperparameters can also have a very large impact on the performances of the models, a Bayesian optimization algorithm [23] is used to perform hyperparameters tuning for all the DNN models.

IV. Results and Discussion

The design parameters space for the 2-dimensional and 3-dimensional examples are tabulated in Table 1. The other design parameters are kept constant such as the dielectric constant, $\epsilon_r = 4.2$, relative permeability, μ = 1, dielectric loss tan = 0.02, metal conductivity, σ=6.8e-7 S/m, and metal thickness = 0.03556 mils. For both examples, the modified adaptive sampling algorithm will be run for 20 iterations. The Z-parameters (Z_{11}, Z_{12}, Z_{13}, and Z_{14}) will be modeled from 3 GHz to 9 GHz, with a step of 0.02 GHz. For both examples, we will generate 100 test samples within the design parameters space using random sampling. The validation and testing performances of each model are measured using the coefficient of determination (R-squared).

A. 2-Dimensional Example

Fig. 7 shows the distributions of the training and validation samples in the 1st iteration, 5th iteration, 10th iteration, and 20th iteration in the 2-dimensional design parameters space, R. The optimal set of hyperparameters for DNN model 1 is found to be as follows: number of hidden feedforward layers = 3 and number of hidden neurons in each layer = [231, 27, 106], and the optimal set of hyperparameters for DNN model 2 is found to be: number of hidden feedforward layers = 3 and number of hidden neurons in each layer = [182, 390, 276]. Table 2 compares the performance of the shallow neural network from the 1st to the 20th iteration of the modified adaptive sampling algorithm with the final DNN model. It can be seen that the validation performance of the surrogate model (shallow neural network) improves when the number of iterations increases. Also, the composite DNN model performs better than all the shallow neural networks, which verifies the hypothesis made earlier. Fig. 8 and Fig. 9 show the actual and predicted Z-parameters of test case 1 ($l = 14.1458$ mm, $h = 0.4433$ mm) and test case 2 ($l = 11.6770$ mm, $h = 0.4048$ mm) respectively.

TABLE I
The Design Parameters Search Space for the 2-Dimensional and 3-Dimensional Examples

Design Parameters	2-dimensional example	3-dimensional example
l (mm)	10-15	10-15
h (mm)	0.4-0.6	0.4-0.6
r	—	0.45-0.55

B. 3-Dimensional Example

Fig. 10 shows the distributions of the training and validation samples in the 1st iteration, 5th iteration, 10th iteration, and 20th iteration in the 3-dimensional design parameters space, R. The optimal set of hyperparameters for DNN model 1 is found to be as follows: number of hidden feedforward layers = 3 and number of hidden neurons in each layer = [31, 146, 190], and the optimal set of hyperparameters for DNN model 2 is found to be: number of hidden feedforward layers = 4 and number of hidden neurons in each layer = [328, 368, 114, 182]. Table 3 compares the performance of the shallow neural network from the 1st to the 20th iteration of the modified adaptive sampling algorithm with the final DNN model. The validation performance of the surrogate model (shallow neural network) improves when the number of iterations increases, which is consistent with the previous example. Also, the improvement in performance for the DNN composite model over the shallow neural network is larger if compared to the previous example because the 3-dimensional modeling problem has a higher degree of non-linearity than the 2-dimensional modeling problem, and DNNs are superior in this regard. Fig. 11 and Fig. 12 show the actual and predicted Z-parameters of test case 1 ($l = 14.8598$ mm, $h = 0.4655$ mm, $r = 0.5334$) and test case 2 ($l = 11.9782$ mm, $h = 0.4107$ mm, $r = 0.4573$) respectively.

V. Conclusion

A DNN based method for power-ground plane modeling has been proposed. The modified adaptive sampling is used to prevent the problem of undersampling or oversampling. Two modeling examples, a square-shaped power ground plane, and a square-shaped power ground plane with a hollow square at the center are used to illustrate the method. Results show that

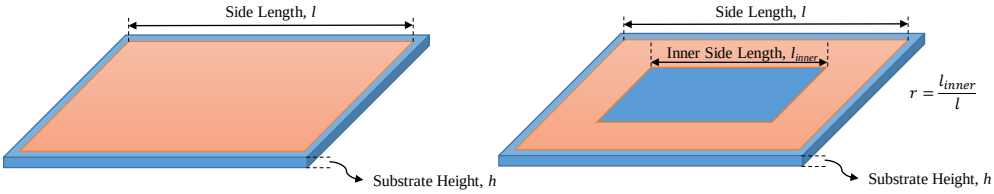

Fig. 2. A square-shaped power ground plane (left) and a square-shaped power ground plane with a hollow square at the center (right)

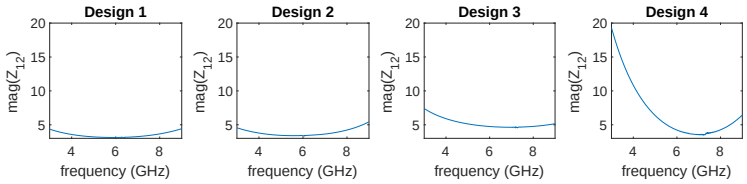

Fig. 3. Z-parameters of four different designs

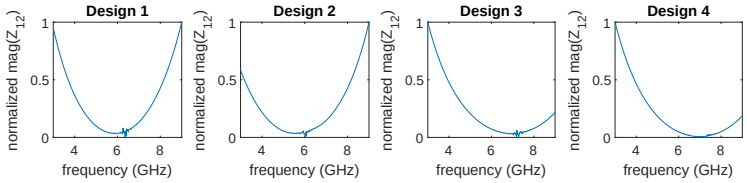

Fig. 4. Z-parameters of four different designs, normalized independently

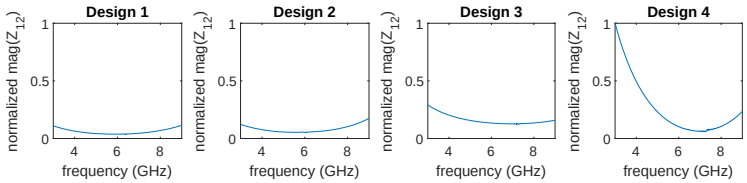

Fig. 5. Z-parameters of four different designs, normalized across all designs

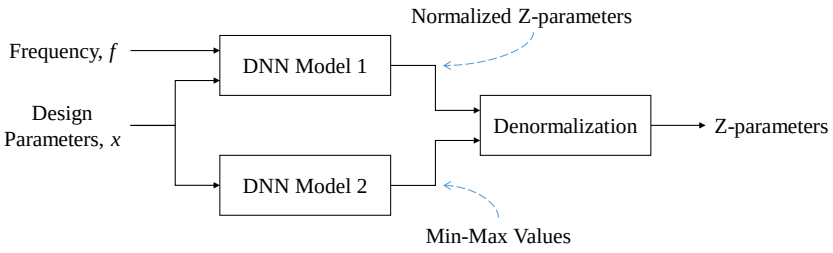

Fig. 6. Structure of the proposed DNN-based composite model for power ground plane modeling

TABLE II
PERFORMANCES OF THE SHALLOW NEURAL NETWORKS IN VARIOUS ITERATIONS OF THE MODIFIED ADAPTIVE SAMPLING ALGORITHM AND THE DNN COMPOSITE MODEL FOR THE 2-DIMENSIONAL EXAMPLE

Model	Validation performance	Testing performance
1st iteration shallow neural network	-3.7323	—
5th iteration shallow neural network	0.9237	—
10th iteration shallow neural network	0.9707	—
20th iteration shallow neural network	0.9851	0.9807
Composite DNN model	0.9949	0.9946

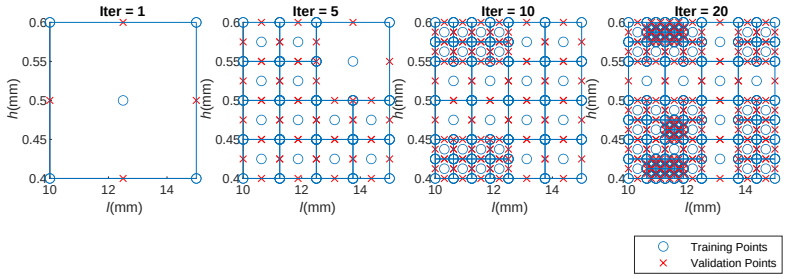

Fig. 7. Distributions of training and validation samples in the 1st iteration, 5th iteration, 10th iteration, and 20th iteration

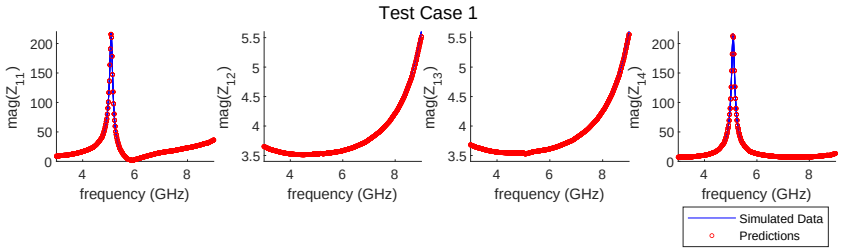

Fig. 8. The magnitudes of Z_{11}, Z_{12}, Z_{13}, and Z_{14} for the 2-dimensional test case 1

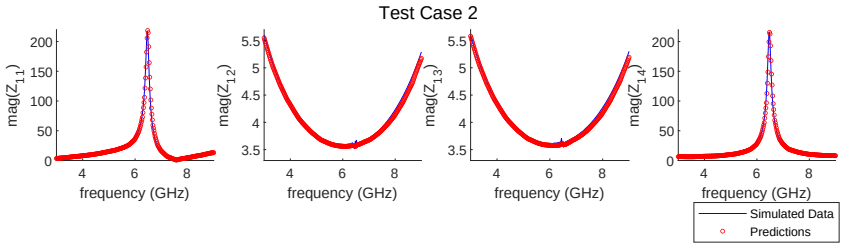

Fig. 9. The magnitudes of Z_{11}, Z_{12}, Z_{13}, and Z_{14} for the 2-dimensional test case 2

TABLE III
PERFORMANCES OF THE SHALLOW NEURAL NETWORKS IN VARIOUS ITERATIONS OF THE MODIFIED ADAPTIVE SAMPLING ALGORITHM AND THE DNN COMPOSITE MODEL FOR THE 3-DIMENSIONAL EXAMPLE

Model	Validation performance	Testing performance
1st iteration shallow neural network	-2.5094	—
5th iteration shallow neural network	0.9162	—
10th iteration shallow neural network	0.9270	—
20th iteration shallow neural network	0.9341	0.9269
Composite DNN model	0.9939	0.9911

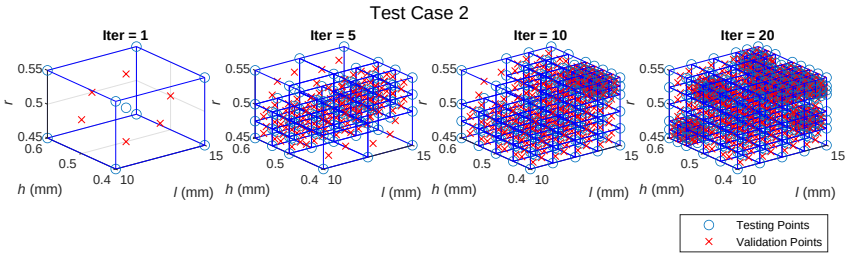

Fig. 10. Distributions of training and validation samples in the 1st iteration, 5th iteration, 10th iteration, and 20th iteration

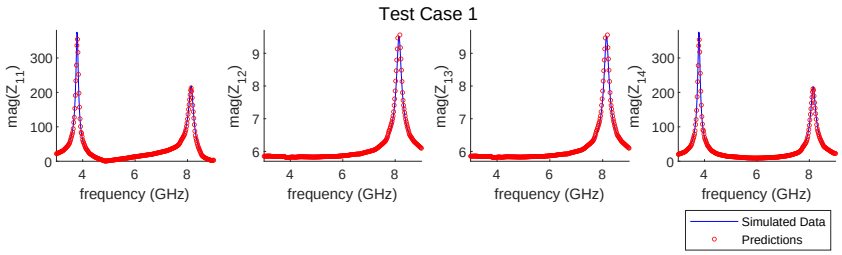

Fig. 11. The magnitudes of Z_{11}, Z_{12}, Z_{13}, and Z_{14} for the 3-dimensional test case 1

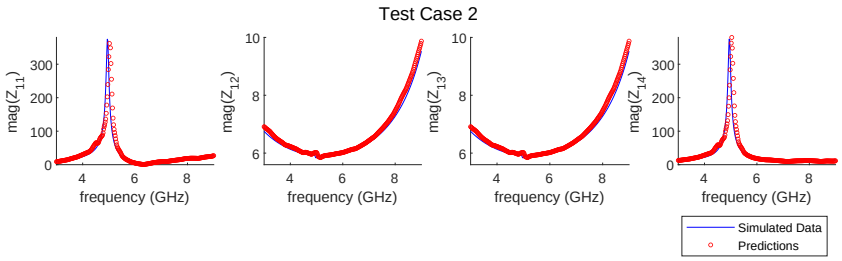

Fig. 12. The magnitudes of Z_{11}, Z_{12}, Z_{13}, and Z_{14} for the 3-dimensional test case 2

the DNN composite model is superior to the shallow neural networks in terms of the prediction accuracy. Nevertheless, despite having lower performances, the shallow neural networks can still be used as the surrogate model during the adaptive sampling process, due to it being much cheaper to train and update.

REFERENCES

[1] S. H. Hall and H. L. Heck, "Advanced Signal Integrity for High-Speed Digital Designs", Hoboken, NJ, USA: Wiley, 2011.

[2] Q. J. Zhang, K. C. Gupta, and V. K. Devabhaktuni, "Artificial neural networks for RF and microwave design - From theory to practice," IEEE Trans. Microw. Theory Tech., vol. 51, no. 4, pp. 1339–1350, 2003. https://doi.org/10.1109/TMTT.2003.809179

[3] H. Kabir, M. Yu, and Q. J. Zhang, "Recent advances of neural network based EM-CAD," Int. J. RF and Microwave CAE, vol. 20, pp. 502-511, Sep. 2010 https://doi.org/10.1002/mmce.20456

[4] M. R. Mohammadi, S. A. Sadrossadat, M. G. Mortazavi and B. Nouri, "A brief review over neural network modeling techniques," in 2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI), pp. 54-57, 2017. https://doi.org/10.1109/ICPCSI.2017.8391781

[5] C. K. Ku, C. H. Goay, N. S. Ahmad, and P. Goh, "Jitter decomposition of high-speed data signals from jitter histograms with a pole–residue representation using multilayer perceptron neural networks," IEEE Transactions on Electromagnetic Compatibility, vol. 62, no. 5, pp. 2227–2237, 2020. https://doi.org/10.1109/TEMC.2019.2936000

[6] V. K. Devabhaktuni and Q. Zhang, "Neural network training-driven adaptive sampling algorithm for microwave modeling," in 2000 30th European Microwave Conference, pp. 1-4, 2000. https://doi.org/10.1109/EUMA.2000.338591

[7] C. H. Goay, A. Abd Aziz, N. S. Ahmad and P. Goh, "Eye diagram contour modeling using multilayer perceptron neural networks with adaptive sampling and feature selection," IEEE Transactions on Components, Packaging and Manufacturing Technology, vol. 9, no. 12, pp. 2427-2441, Dec. 2019. https://doi.org/10.1109/TCPMT.2019.2938583

[8] T. Lu, J. Sun, K. Wu and Z. Yang, "High-speed channel modeling with machine learning methods for signal integrity analysis," IEEE Transactions on Electromagnetic Compatibility, vol. 60, no. 6, pp. 1957-1964, Dec. 2018. https://doi.org/10.1109/TEMC.2017.2784833

[9] C. H. Goay, N. S. Ahmad, and P. Goh, "Transient simulations of high-speed channels using CNN-LSTM with an adaptive successive halving algorithm for automated hyperparameter optimizations," IEEE Access, vol. 9, pp. 127 644–127 663, 2021. https://doi.org/10.1109/ACCESS.2021.3112134

[10] J. Jin, C. Zhang, F. Feng, W. Na, J. Ma and Q. Zhang, "Deep neural network technique for high-dimensional microwave modeling and applications to parameter extraction of microwave filters," IEEE Transactions on Microwave Theory and Techniques, vol. 67, no. 10, pp. 4140-4155, Oct. 2019. https://doi.org/10.1109/TMTT.2019.2932738

[11] J. Jin, F. Feng, J. Zhang, S. Yan, W. Na and Q. Zhang, "A novel deep neural network topology for parametric modeling of passive microwave components," IEEE Access, vol. 8, pp. 82273-82285, 2020. https://doi.org/10.1109/ACCESS.2020.2991890

[12] L. Kouhalvandi, O. Ceylan and S. Ozoguz, "Automated deep neural learning-based optimization for high performance high power amplifier designs," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 67, no. 12, pp. 4420-4433, Dec. 2020. https://doi.org/10.1109/TCSI.2020.3008947

[13] H. Mhaskar, Q. Liao and T. Poggio, "When and why are deep networks better than shallow ones?", in Proc. Thirty–First AAAI Conference on Artificial Intelligence (AAAI-17), pp. 2343-2349, 2017.

[14] S. Liang and R. Srikant, "Why deep neural networks for function approximation?", in Proc. 5th Int. Conf. Learn. Represent. (ICLR), pp. 1-17, Apr. 2017. https://doi.org/10.48550/arXiv.1610.04161

[15] F. Emmert-Streib, Z. Yang, H. Feng, S. Tripathi, and M. Dehmer, "An introductory review of deep learning for prediction models with big data," Frontiers Artif. Intell., vol. 3, pp. 1–23, Feb. 2020. https://doi.org/10.3389/frai.2020.00004

[16] K. Pasupa and W. Sunhem, "A comparison between shallow and deep architecture classifiers on small dataset," in 2016 8th International Conference on Information Technology and Electrical Engineering (ICITEE), pp. 1-6, 2016. https://doi.org/10.1109/ICITEED.2016.7863293

[17] J. H. Lee, "A novel meander split power/ground plane reducing crosstalk of traces crossing over," Electronics, vol. 8, no. 9, p. 1041, Sep. 2019.

[18] K. Shringarpure, "Printed circuit board power distribution network modeling, analysis and design, and, statistical crosstalk analysis for high speed digital links," Ph.D. dissertation, Dept. Elect. Comput. Eng., Missouri Univ. Sci. Technol., Rolla, MO, USA, 2015.

[19] W. D. Becker *et al.*, "Modeling, simulation, and measurement of mid-frequency simultaneous switching noise in computer systems," IEEE Transactions on Components, Packaging, and Manufacturing Technology, vol. 21, no. 2, pp. 157-163, May 1998. https://doi.org/10.1109/96.673703

[20] Altera Corporation, Appl. Note AN574, "Printed Circuit Board (PCB) Power Delivery Network (PDN) Design Methodology," May 2009.

[21] M. Abadi et al., "TensorFlow: A system for large-scale machine learning," in 12th USENIX Symposium on Operating Systems Design and Implementation, 2016, pp. 265–283. https://doi.org/10.48550/arXiv.1605.08695

[22] D. Kingma, J. Ba, "ADAM: A method for stochastic optimization", in International Conference on Learning Representations (ICLR), 2015, pp. 11-15. https://doi.org/10.48550/arXiv.1412.6980

[23] J. Jimenez and J. Ginebra, "pyGPGO: Bayesian optimization for python", Journal of Open Source Software, vol. 2, no. 19, pp. 431, 2017. https://doi.org/10.21105/joss.00431