# Surrogate Modeling with Complex-valued Neural Nets and its Application to Design of sub-THz Patch Antenna-in-Package

Oluwaseyi Akinwande*, Osama Waqar Bhatti*, Kai-Qi Huang*, Xingchen Li*, Madhavan Swaminathan*‡
*School of Electrical and Computer Engineering, Georgia Institute of Technology, USA
‡School of Electrical Engineering and Computer Science, Pennsylvania State University, USA

2023 IEEE/MTT-S International Microwave Symposium - IMS 2023 | 979-8-3503-4764-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/IMS37964.2023.10187990

***Abstract*— In this paper, we propose a surrogate model for both forward and inverse modeling with complex-valued neural networks. The complex domain offers the benefits of higher functionality and better representation. To that end, we propose a deep complex dense network (ℂDNet) by introducing complex dense blocks built with fully-connected layers that support complex operations. We further propose an inverse optimization objective that minimizes the modeling error while optimizing the design space parameters that achieve the target specifications. We apply our proposed approach for the design of a sub-THz patch antenna-in-package operating at 140 GHz frequency band.**

***Keywords*— surrogate modeling, inverse design, complex-valued neural networks, antenna-in-package, sub-terahertz, design space exploration.**

## I. Introduction

Predominantly, neural networks are trained in the real-valued domain $\mathbb{R}$. It has been applied with great success in many applications such as speech recognition, computer vision, autonomous driving, image identification, drug discovery, stock market trading, natural language processing, to name a few. However, complex-valued neural networks bring a whole new paradigm to machine learning. The absence of such methods represents a gap in machine-learning toolbox since complex numbers are not just an abstract mathematical construct, they exist in physical phenomena. Complex numbers have been found to exist in physical signals such as magnetic resonance imaging (MRI) data, electromagnetic (EM) scattering parameters, x-ray crystallography, computed tomography (CT) scan, and so on. Complex-valued neural networks have been applied in these areas and to real-valued signals as well with results comparable to their real-valued counterparts.

There are several benefits to using complex-valued neural networks, which we highlight as follows: First, they provide a higher functionality by incorporating the phase information since learnable weights do not just change amplitude as in the real-valued case, but can be rotated too in the complex-valued case. Next, complex numbers offer a richer set of numbers than the normal real numbers in neural networks. This implies that complex-valued neural networks have a more compact data representation with a mapping represented as $f(z) : \mathbb{C}^N \mapsto \mathbb{C}^M$, where $N, M$ are the number of input and output dimensions, respectively. In contrast, the real-valued neural networks have a less prudent data representation by using only the magnitude component or stacking the real and imaginary parts and handling them as real-valued data with a mapping represented as $f(z) : \mathbb{R}^{2N} \mapsto \mathbb{R}^M$ [1].

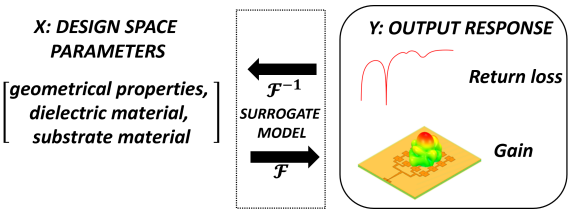


Fig. 1. Surrogate model that offers a custom solution.

Furthermore, in the field of classification with deep neural networks, real-valued neural networks have made great strides, although, a single perceptron still has limitations. For example, a single perceptron can only learn linearly separable Boolean functions such as `AND` and `OR` but not linearly nonseparable functions such as `XOR` [2]. Conversely, a single complex-valued perceptron employs orthogonal decision boundaries to solve the `XOR` problem [2].

In the foregoing, we exploit the advantages offered by complex representations to build a surrogate model for both forward and inverse modeling. Typically, in early-stage prototyping, a designer comes up with an initial design with several parameters in the design space. If the design does not satisfy the target specifications, the designer has to do another iteration and gauge the output response with the target specifications. Usually, the designer explores several parameters and laboriously goes through multiple iterations in order to satisfy the target specifications. This is the forward modeling process. Inverse modeling, on the other hand, starts from the target specifications and generates the circuit parameters that satisfy the target in one fell swoop.

In this paper, we present a new end-to-end learning with complex-valued data targeted at broadband $S-$parameter modeling. We propose a deep complex dense network (ℂDNet) by introducing complex dense blocks built with fully-connected layers that support complex operations. We also propose an inverse optimization objective that minimizes the modeling error while optimizing the design space parameters that achieve the target specifications. We test our proposed approach on the design of a sub-THz patch antenna-in-package.

## II. Complex Building Blocks

Having emphasized the importance of complex-valued neural networks, the question that arises is how to set it up. We employ some innovative approaches from existing literature to form the building blocks needed for a complex-valued neural network similar to its real-valued counterpart. In the following discourse, let $z = a + jb$ represent a complex-valued input, where $j = \sqrt{-1}$.

### A. Complex Dense Block

The complex dense block introduces feed-forward connections from one layer to the next. This encourages feature reusability and strengthens information propagation through the network [3]. Let weight $w = w_{\mathcal{R}} + jw_{\mathcal{I}}$, the complex dense operator performs the complex operations:

$$w * z = (a * w_{\mathcal{R}} - b * w_{\mathcal{I}}) + j(a * w_{\mathcal{I}} + b * w_{\mathcal{R}}), \quad (1)$$

as shown in Fig. 2. In these notations, $*$ denotes the complex dense operation, and the subscripts $\mathcal{R}$ and $\mathcal{I}$ denote the real and imaginary parts of a complex-valued entity, respectively.

### B. Complex Activations

As with their real-valued counterparts, complex-valued activations are used to achieve nonlinearity. Designing a complex activation is challenging due to the constraints postulated in Liouville's theorem [4] [1]. We employ fully complex activations, and split-activations where the non-linearity is applied separately on the real and imaginary parts. Examples include:

$$\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \quad (2)$$

$$\mathbb{C}\text{ReLU}(z) = \text{ReLU}(a) + j\text{ReLU}(b) \quad (3)$$

$$\mathbb{C}\text{LeakyReLU}(z) = \text{LeakyReLU}(a) + j\text{LeakyReLU}(b) \quad (4)$$

(2), (3) and (4) are the complex hyperbolic tangent, split-complex rectified linear unit, and split-complex leaky rectified linear unit, respectively.

[1]Liouville's theorem states that a complex-valued function which is bounded and analytic everywhere (i.e., a function that is differentiable at every point ), is constant. In other words, a function that is bounded and analytic everywhere in the complex plane is not a suitable complex activation function [1].

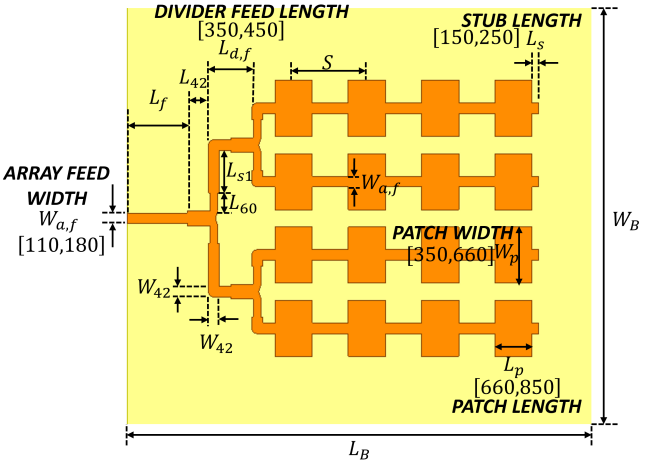


Fig. 3. Patch antenna-in-package [5]. All dimensions in $\mu$m.

### C. Complex Residual Blocks

The complex residual blocks enable skip connections which, as the name suggests, skip some of the layers in the neural network, and add the original input back to the output feature map obtained by passing the input through one or more complex dense layers. This is relevant for preserving contextual information. Furthermore, skip connections prevent the vanishing gradient problem, by directly propagating gradients between layers. Consider a complex neural network block that provides a mapping $T(z)$ from the input layer to the output layer. The residual is

$$R(z) = T(z) - z. \quad (5)$$

(5) can be re-arranged to form

$$T(z) = R(z) + z, \quad (6)$$

which first applies an identity mapping to $z$, then it performs element-wise complex addition.

## III. Application: Inverse Design of Sub-THz Patch Antenna-in-Package

We apply the proposed deep complex dense network ($\mathbb{C}$DNet) for the design of a sub-THz patch antenna array

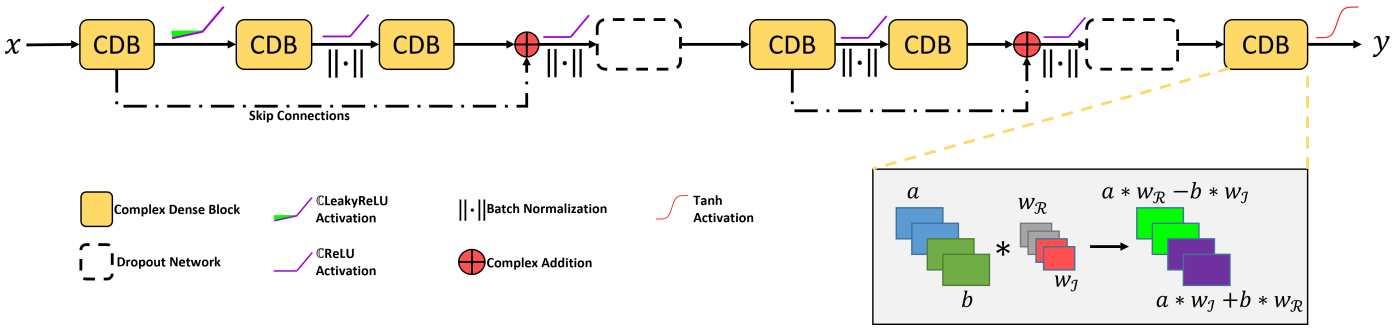


Fig. 2. Proposed deep complex dense network ($\mathbb{C}$DNet). In inverse modeling, we backpropagate the gradients of the trained $\mathbb{C}$DNet to update its input parameters and minimize the cost function of the measure of performance. The set of patch array input parameters that minimize the cost function is the inverse solution. $x$ : design space parameters, $y$ : target specifications.

operating at 140 GHz. This is a 4-by-4 rectangular low-profile antenna array (see Fig. 3), and has applications in 5G/6G wireless technologies. Its stack-up is made up of microstrip structure built on top of glass interposer with ground planes beneath. Its frequency response is the $S_{11}$. The objective here is to build a fast surrogate model that enables the designers to (1) simulate their designs to ensure they are within a few dB of the target specifications of $S_{11}$ from $130.1-150.05$ GHz, and (2) obtain the patch antenna design parameters that correspond to a given specification of the $S_{11}$ in the target band. Using Latin Hypercube Sampling, we determine 2500 samples and extract their $S$-parameters with Ansys HFSS [6]. We split the data into train and test sets.

### A. Forward Model

The goal of the forward model is to train the deep complex dense network ($\mathbb{C}$DNet) to learn the forward mapping between the patch array design space $x$ and output response $y$ (i.e., the $S_{11}$). We build on top of the complex-valued neural net in [7]. The proposed model is illustrated in Fig. 2. The deep complex dense network ($\mathbb{C}$DNet) is constructed using 6 complex dense blocks (see section II) with skip connections between them for preserving contextual information. Each complex dense block contains one hidden layer of 256 neurons. We apply $10\%$ dropout regularization after the skip connections. Complex rectified linear unit ($\mathbb{C}$ReLU) activation functions and batch normalization layers are used between the complex dense blocks except for the first complex dense block which only has a $\mathbb{C}$LeakyReLU activation. During training, the end-to-end neural net takes input $x$ with 5 patch array design parameters, and propagates the information through the network to generate $y$ (i.e., the $S_{11}$). We train with an $\ell_2$-supervised loss given by

$$\mathcal{L} = \mathbb{E}_{x,y}\left[||\hat{y}_{\mathcal{R}} - y_{\mathcal{R}}||_2^2 + ||\hat{y}_{\mathcal{I}} - y_{\mathcal{I}}||_2^2\right], \tag{7}$$

where $\hat{y}$ is the predicted $S_{11}$.

### B. Inverse Optimization

The use of optimization in inverse modeling allows us to adjust the patch array parameters, calibrate, and provide the best solutions for our design. By identifying some objective and a measure of performance of the system, we arrive at optimal solutions for our design. The objective depends on the parameters we specify. The goal is to find the parameters that optimize the objective. Often, parameters are constrained in some way, which we must take into account and judiciously optimize so as give physically realizable results for the patch array.

In designing our sub-THz patch antenna array, the ideal goal is to have no reflected signal (i.e., $|S_{11}| = 0$) in specific frequency bands, and to have all of the signal reflected (i.e., $|S_{11}| = 1$) outside these target bands. We can define the objective as the $\ell_2$-norm of the difference between the ideal $|S_{11}|$ (i.e., $y^*$) and that delivered by the forward model (i.e. $\hat{y}(x)$):

$$\mathcal{J}(\hat{y}) = \|\hat{y}(x) - y^*\|_2^2. \tag{8}$$

**Algorithm 1:** Inverse optimization

**Input:** Initialization $x^{(0)} \in dom(g)$, trained model $g$ with the set of all network parameters $\theta$, target band $B^*$, learning rate $\lambda$

**Output:** estimated $x$

**for** $k = 0, 1, 2, ...,$ *until convergence,* **do**

$$\hat{y}^{(k)} = g(x^{(k)}, \theta)$$

$$\mathcal{J}(\hat{y}^{(k)}) = \sum_{i:f_i \in B^*} |\hat{y}_i^{(k)}|^2 + \sum_{i:f_i \notin B^*} (|\hat{y}_i^{(k)}| - 1)^2$$

$$\Delta x^{(k)} = -\frac{\partial \mathcal{J}(\hat{y}^{(k)})}{\partial \hat{y}^{(k)}} \frac{\partial \hat{y}^{(k)}}{\partial \theta} \frac{\partial \theta}{\partial x^{(k)}}$$

Update: $x^{(k+1)} \leftarrow x^{(k)} + \lambda \Delta x^{(k)}$

Given a target band $B^*$, we can re-write the objective in (8) as:

$$\mathcal{J}(\hat{y}) = \sum_{i:f_i \in B^*} |\hat{y}_i(x)|^2 + \sum_{i:f_i \notin B^*} (|\hat{y}_i(x)| - 1)^2. \tag{9}$$

We will like to minimize the objective in (9) parameterized by the patch array parameters $x$ and the set of all network parameters $\theta$. We use gradient descent to learn the optimal parameters that give the minimum cost in (9). The optimized patch array parameters $x$ is the inverse solution. The outline of the inverse optimization method is shown in Algorithm 1.

### C. Evaluation Metric

We evaluate our resulting antenna with two metrics: the pass-band Intersection-over-Union (IoU) metric introduced in [7], and the return loss passband. The IoU metric is a method to quantify the percentage overlap between the target band $B^*$ and our prediction passband $\hat{B}$.

$$\text{IoU} = \frac{B^* \cap \hat{B}}{B^* \cup \hat{B}}, \tag{10}$$

where the passband $B$ is, generally, the region where the $|S_{11}|$ is lower than $-10$ dB in the resonant band, i.e. $B = \{[f_L, f_H] : |S_{11}| < -10\text{dB}\}$.

## IV. Results

We present the results from both the forward model training and the inverse optimization. We perform inference for the forward model by taking a random sample from the test set. Fig. 4 shows the results obtained from the forward modeling. We compare the real, imaginary, and magnitude components of the $S_{11}$ obtained from the forward model with those obtained from the EM simulator. We find that there is a close correlation of the output responses from the forward model and the EM simulator.

Next, we display the inverse optimization results in Fig. 5. Given an arbitrary target band $B^* = [138, 142]$ GHz, we optimize to find the patch array input parameters that best achieves this target within the given constraints of the patch array input parameters (see Fig. 3 for their respective

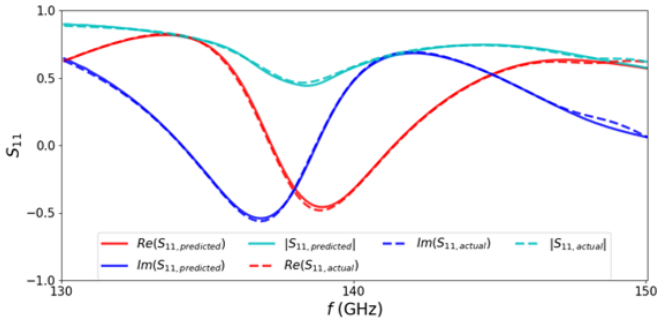


Fig. 4. Forward modeling predictions showing $S_{11}$ for the patch antenna array with the trained complex-valued neural network model (indicated with solid lines), compared with the EM simulation (indicated with dashed lines) for a random design tuple in the test set. The real, imaginary and magnitude components of the $S_{11}$ response are shown in red, blue and cyan, respectively.

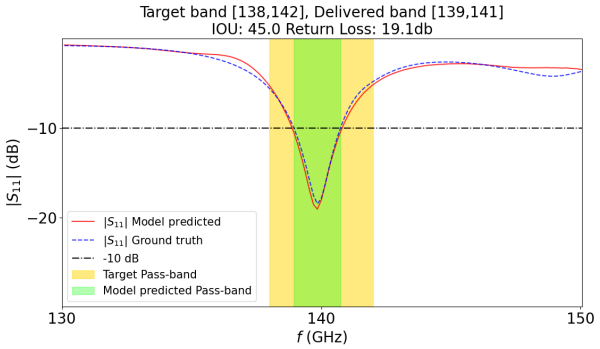


Fig. 5. Inverse optimization results illustrating the target band (indicated with yellow) and the predicted passband (indicated with green). The optimization yielded an inverse solution of $\{502.7, 789.8, 176.7, 210.3, 340.4\}$ $\mu$m for the patch array parameters. For this solution, the model predicts the return loss in the solid red line, while the EM simulation of the inverse solution results in the return loss in the dashed blue line.

ranges). Fig. 5 illustrates that the algorithm was able to deliver an IoU of $45\%$ and a return loss of 19.1 dB at the 140 GHz resonance, for a prediction of $\{502.7, 789.8, 176.7, 210.3, 340.4\}$ $\mu$m corresponding to the patch array input parameters $\{W_p, L_p, W_{a,f}, L_s, L_{d,f}\}$.

A summary of the performance of the proposed surrogate model is provided in Table 1. The metric to assess the numerical accuracy of the forward model is chosen as the mean absolute error (MAE) in decibels, between the actual response $y$ and the predicted response $\hat{y}$, taken over all the frequency points in the response, given by:

$$\Delta = \frac{1}{r}\sum_{i=1}^{r} \left| 20\log_{10}|y_i| - 20\log_{10}|\hat{y}_i| \right|. \qquad (11)$$

In this working example, it took ∼617 CPU hours in collecting the training data and ∼1.8 minutes to train the $\mathbb{C}$DNet model. Once trained, the $\mathbb{C}$DNet can generate 2500 frequency responses in ∼2.5 seconds.

## V. Conclusion

We present both forward and inverse modeling of RF systems by using complex-valued neural networks. The forward model takes the input parameters of the RF model, propagates the information through a series of building blocks with complex operations and generates the complex-valued output response. However, in the inverse modeling, we propose a well-defined objective as a measure of performance of the RF system and optimize this objective using gradient descent to obtain the optimal input parameters. This surrogate model has the capability of reducing design cycle time and it gives the designer a quick prototype.

Table 1. Performance Summary of the Proposed Surrogate Model

| Design parameters | Frequency points | Train error (for 2400 samples) | Inference error (for 100 samples) |
|---|---|---|---|
| 5 | 134 | 0.641 dB | 0.357 dB |

† All programming is performed with PyTorch [8] on a Windows desktop with Intel® Core™ i7-10700 CPU @ 2.90 GHz and 32 GB RAM.


## Acknowledgment

This material is based upon work supported by the National Science Foundation under Grant No. CNS 2137259 - Center for Advanced Electronics through Machine Learning (CAEML).


## References


[1] C. Lee, H. Hasegawa, and S. Gao, "Complex-valued neural networks: A comprehensive survey," *IEEE/CAA Journal of Automatica Sinica*, vol. 9, no. 8, pp. 1406–1426, 2022.

[2] N. T., "Solving the xor problem and the detection of symmetry using a single complex-valued neuron," *Neural Networks*, 2003.

[3] M. A. Dedmari, S. Conjeti, S. Estrada, P. Ehses, T. Stöcker, and M. Reuter, "Complex fully convolutional neural networks for MR image reconstruction," *CoRR*, vol. abs/1807.03343, 2018. [Online]. Available: http://arxiv.org/abs/1807.03343

[4] J. Liouville, "Lecons sur les fonctions doublement périodiques faites en 1847. première partie. théorie générale." *Journal für die reine und angewandte Mathematik*, vol. 88, pp. 277–310, 1879. [Online]. Available: http://eudml.org/doc/148443

[5] K.-Q. Huang and M. Swaminathan, "Antennas in glass interposer for sub-thz applications," in *2021 IEEE 71st Electronic Components and Technology Conference (ECTC)*, 2021, pp. 1150–1155.

[6] Ansys Inc., "Ansys HFSS," `https://www.ansys.com/`, 2022.

[7] G. Zhang, H. He, and D. Katabi, "Circuit-GNN: Graph neural networks for distributed circuit design," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 97. PMLR, 09–15 Jun 2019, pp. 7364–7373. [Online]. Available: https://proceedings.mlr.press/v97/zhang19e.html

[8] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Z. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," *CoRR*, vol. abs/1912.01703, 2019. [Online]. Available: http://arxiv.org/abs/1912.01703