Bayesian Learning for Uncertainty Quantification, Optimization, and Inverse Design

Madhavan Swaminathan, *Fellow, IEEE*, Osama Waqar Bhatti, *Graduate Student Member, IEEE*, Yiliang Guo, *Graduate Student Member, IEEE*, Eric Huang, and Oluwaseyi Akinwande, *Graduate Student Member, IEEE*

***Abstract*— Design of microwave circuits require extensive simulations, which often take significant computational time due to design complexity. This can be addressed through neural networks (NNs) that provide predictive capability. Predictions often come with uncertainties that need to be quantified. Moreover, optimization and inverse designs are better done using probabilities. This article describes the use of Bayes theorem and machine learning (ML) for solving complex microwave design problems.**

***Index Terms*— Bayes theorem, Gaussian processes (GPs), invertible neural networks, neural networks (NNs).**

I. Introduction

BAYES theorem had an early beginning in 1740 through the Bayes' rule proposed by an amateur mathematician Rev. Thomas Bayes. This was discovered later by Pierre Simon Laplace in 1774 and worked on for the next 40 years. The Bayes' rule came into prominence many years later when Alan Turing used it to break Enigma, during the second world war [1].

So, what is Bayes theorem and why apply it to microwave design? Bayes theorem is based on subjective probability as opposed to objective probability practiced by frequentists. In mathematical form it can be written as follow:

$$P(H|E) = \frac{P(E|H)P(H)}{P(E)} \tag{1}$$

where H is the hypothesis, E the evidence, $P(H|E)$ is the conditional probability of the hypothesis when the evidence is considered (posterior), $P(E|H)$ is the conditional probability of the evidence given the hypothesis is true (likelihood), $P(H)$ is the probability of the hypothesis before the evidence is considered (prior), and $P(E)$ is the probability of the evidence under any circumstance (marginal probability) given as follow:

$$P(E) = P(E|H)P(H) + P(E|H')P(H') \tag{2}$$

Manuscript received 17 March 2022; revised 5 June 2022 and 26 August 2022; accepted 30 August 2022. Date of publication 6 October 2022; date of current version 4 November 2022. This work was supported in part by the National Science Foundation through the Center for Advanced Electronics through Machine Learning (CAEML) under Grant CNS 16-2137259 and in part by the Georgia Tech 3D Systems Packaging Research Center (PRC). *(Corresponding author: Osama Waqar Bhatti.)*

The authors are with the Department of Electrical and Computer Engineering, Materials Science and Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: osamawaqarbhatti@gatech.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TMTT.2022.3206455.

Digital Object Identifier 10.1109/TMTT.2022.3206455

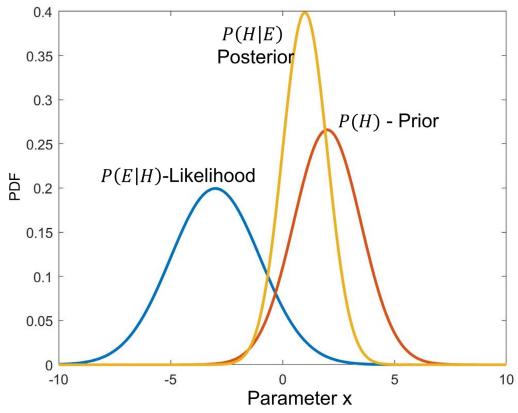


Fig. 1. Updating prior to obtain the posterior after seeing the likelihood according to (1). Here, prior $P(H)$ is assumed to be a Gaussian distribution with large variance. After observing the evidence based on the hypothesis, the prior is updated to the posterior $P(H|E)$ with less variance, and hence with more confidence [2]. Note: This is shown for illustration purposes only since the distributions can be arbitrary.

where $P(H') = 1 - P(H)$ is the probability of the hypothesis not being true and $P(E|H')$ is the conditional probability of the evidence when the hypothesis is untrue. The denominator term in (1) is generally a normalizer to ensure that the result in (1) is always a probability that is bounded between 0 and 1.

As an example, in Fig. 1, the prior is assumed to be a Gaussian distribution with large variance, where the random variable is the parameter x. By making use of the likelihood and the prior, the posterior is computed with less variance. A smaller variance translates into a better confidence in the prediction, or in other words smaller uncertainty. Being subjective Bayes theorem allows for guesses where the prior distribution can be assumed, making it powerful and applicable in several areas.

In complex microwave designs, the parameter space can be large leading to high-dimensional problems [3]. As a result, the probabilities in (1) become intractable and therefore cannot be computed analytically. Therefore, Bayes theorem needs to be integrated into a computational environment for the estimation of probabilities. In machine learning (ML), the goal is to learn the mapping between the input and output parameters to be able to predict the output for a given input. This can be enabled by combining Bayes theorem with ML, also called Bayesian learning in this article. This approach can be used

to develop a surrogate model with confidence bounds (uncertainty quantification), used to solve non-convex optimization problems, and address nonuniqueness in inverse problems, as described in Fig. 2 and discussed in Sections II–IV. Due to high frequencies, microwave designs rely on circuit and electromagnetic (EM) simulations for achieving success. Due to high dimensionality and design complexity, simulators are time consuming and therefore their use as part of the optimization process has only had limited success. With the advent of artificial neural networks (ANNs), the large computational time can be addressed through a onetime training process where the neural network (NN) [4] can be used to map between the input and output parameters. NNs allow for accurate linear and nonlinear mapping between the parameters, where the architecture of the networks can be constructed based on the data patterns being learned. However, such NNs tend to produce a discrete output for a set of input parameters. Like any model, NNs are prone to errors and therefore capturing these errors around the predictions become necessary. In optimization, the objective is to converge to the global optimum. This can become extremely difficult if the optimization is being conducted on a non-convex surface. With Bayes theorem, the problem can be reposed such that the computed probabilities provide a direction for the optima as opposed to an exact solution at intermediate points and therefore this property can be used to rapidly find the global optimum. Finally, solutions to inverse problems provide an elegant method for creating designs based on the output objectives. However, these problems are often-times ill-posed. We address these three areas in the context of microwave design in this article. Our hope is that this article will get more researchers and designers interested and motivated to work in this upcoming area of ML.

This article is organized as follows. We discuss, in detail, the three scenarios of uncertainty quantification, optimization and inverse design, in the subsequent Sections II–IV, respectively, with corresponding examples from literature and recent advances. In Section V, we provide conclusion.

II. Uncertainty Quantification

NNs are deterministic and therefore provide a point estimate. These networks are generally considered to be always correct, implying that the predictions have no errors in them. This obviously cannot be true since the model parameters and data samples used in the NNs can generate errors. Hence quantifying these uncertainties in the predictions become necessary. We describe two networks here for uncertainty quantification namely the Bayesian NN (BNN) and the other constructed using the Gaussian process (GP).

A. Bayesian NN (BNN)

We start by defining a NN before to converting it to the BNN. As an example, consider the spectral transposed convolution NN (S-TCNN) described in [5] which is useful for the prediction of frequency responses. As the name implies, S-TCNN transposes a Convolution NN (CNN) such that low dimensional features can be mapped to high-dimensional frequency responses. The input parameters are first mapped to a latent space through a fully connected NN. The features are then learned using transposed 1-D convolution where the output y is related to input x by the relation as follow:

$$\begin{aligned} y &= f(\boldsymbol{h} *^T \boldsymbol{x}) \\ &= f(\boldsymbol{H^T x}) \end{aligned} \tag{3}$$

where $f(\cdot)$ is the non-linear activation function, $h = [w_1, w_2, \ldots, w_k]^T$ is the matrix convoluted with input x, $*$ stands for the convolutional operator while $*^T$ stands for the transposed convolutional operator and

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

$$\boldsymbol{H^T} = \begin{bmatrix} w_1 & 0 & \cdots & 0 \\ w_2 & w_1 & \ddots & \vdots \\ \vdots & w_2 & \ddots & 0 \\ w_k & \vdots & \ddots & w_1 \\ 0 & w_k & \ddots & w_2 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & w_k \end{bmatrix}$$

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}. \tag{4}$$

The 1-D kernel has k weights (w_k) that are slid across the flattened frequency response to learn the features such as resonance behavior, sharpness of the peaks etc. This procedure is illustrated in Fig. 3. S-TCNN has been shown to predict the output frequency response accurately with an output dimensionality exceeding $1M$, where each frequency point is considered a dimension. Causality and passivity of the data can be addressed as part of S-TCNN as well, as described in [6]. We will assume here that S-TCNN (or any other NN) is used to predict the frequency response of a microwave structure. The objective is to quantify the uncertainty around these predictions.

Let W and b denote the weights and bias of a fully connected NN. For an input X, the output Y can be written as follow:

$$Y = f(WX + b) \tag{5}$$

where $f(\cdot)$ is the non-linear activation function.

Using Bayes' theorem from (1) the nonlinear mapping between X and Y can be written in the form as follow:

$$p(\theta|X, Y) = \frac{p(Y|X, \theta)p(\theta)}{p(Y|X)} \tag{6}$$

where $p(\theta)$ is the prior, $p(Y|X, \theta)$ is the likelihood and $p(Y|X)$ is the normalizer, also called model evidence. In (6), $\theta = \{\boldsymbol{W}, \boldsymbol{b}\}$ represents the parameters that need to be learned.

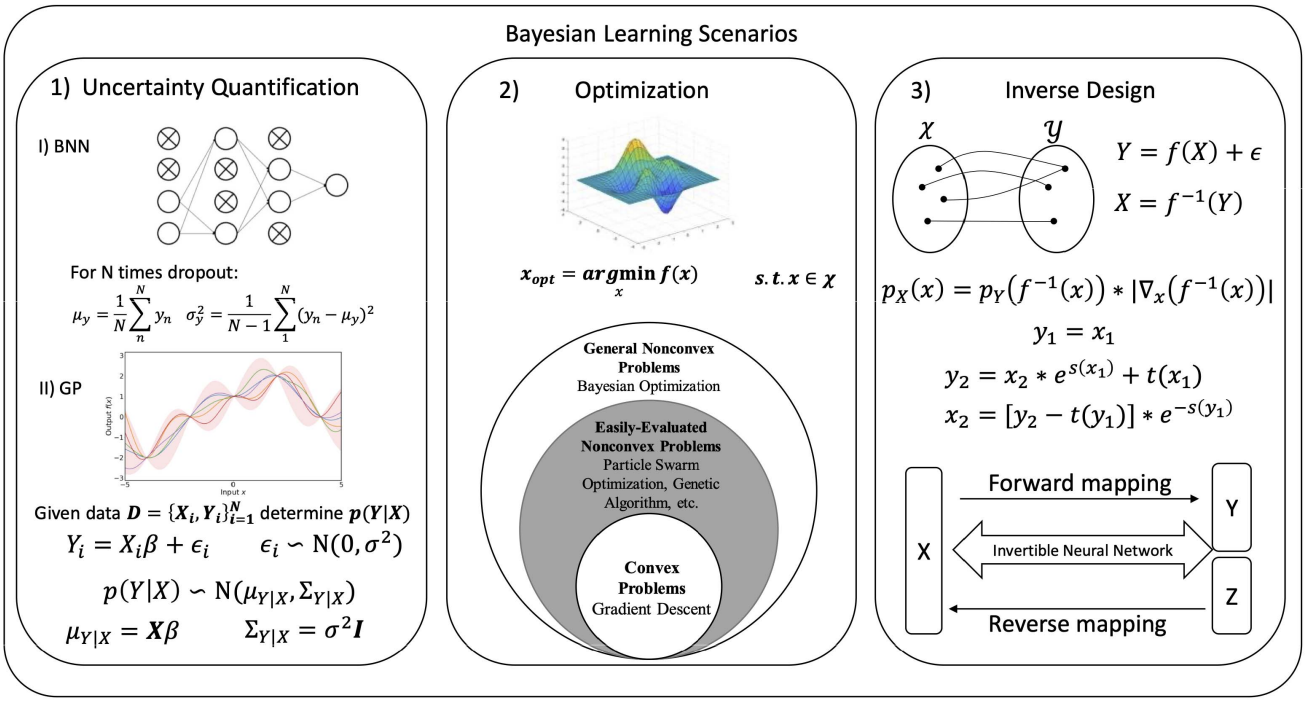


Fig. 2. Bayesian learning scenarios. (a) Uncertainty quantification. (b) Optimization. (c) Inverse design.

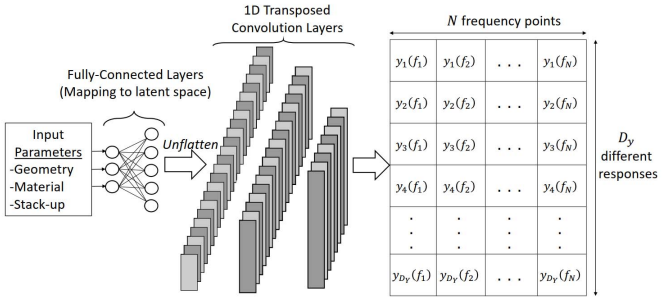


Fig. 3. S-TCNN architecture [7].

Using (6), the output y^* given a new observed input vector x^* can be written as follow:

$$p(y^*|x^*, \mathcal{D}) = \int p(y^*|x^*, \theta)p(\theta|\mathcal{D})d\theta \tag{7}$$

where $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{Y}\}$ represents the dataset. This process is referred to as inference. Computing the full posterior distribution over all possible parameter settings is called Bayesian learning. The integration in (7) can be analytically calculated for some simple models, such as Bayesian linear regression. However with more complicated models, the parameter posterior is analytically intractable. We therefore approximate the posterior distribution for the model parameters via variational inference (VI). The goal of VI is to learn a fast-to-compute approximate distribution, $q(\theta)$, which is as close as possible to the intractable parameter posterior. This can be achieved by minimizing the Kullback–Leiber (KL) divergence [8] between the two distributions as follow:

$$\begin{aligned}\mathcal{L} &= \mathrm{KL}(q(\theta)\|p(\theta|\mathcal{D})) \\ &= \int q(\theta)\log\frac{q(\theta)}{p(\theta|\mathcal{D})}d\theta.\end{aligned} \tag{8}$$

KL divergence minimization is also equivalent to maximizing the evidence lower bound by applying Jensen's inequality [9] as follow:

$$\begin{aligned}\mathcal{L} &= \int q(\theta)\log p(Y|X,\theta)d\theta - \mathrm{KL}(q(\theta)\|p(\theta)) \\ &\leq \log p(Y|X)\end{aligned} \tag{9}$$

where $p(\theta)$ is the prior distribution over NN parameters. The first term in (9) is approximated using Monte Carlo (MC) integration. The second term in (9) can be calculated using the closed-form KL divergence between the true prior and its approximation. This works as an Occam Razor [10] term and penalizes the complexity of $q(\theta)$ [11]. Through the training process, the inference equation in (7) can be rewritten as an approximation in the form [7] as follow:

$$q(y^*|x^*, \mathcal{D}) = \int p(y^*|x^*, \theta)q(\theta|\mathcal{D})d\theta. \tag{10}$$

We still need to choose the type of $q(\cdot)$ to train the models. If we choose a Bernoulli distribution [12] for $q(\cdot)$, this forms the basis for Bayesian dropout which can be used to establish confidence bounds. The Bayesian dropout process is described in Fig. 4 where the neurons within a network are dropped randomly during each inference. We make use of the S-TCNN to implement the Bayesian dropout process.

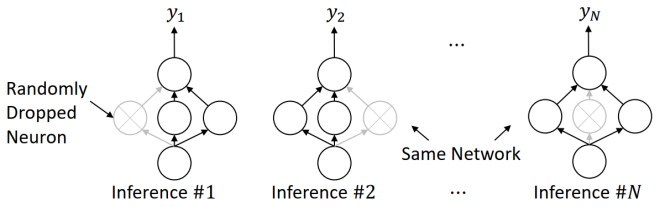


Fig. 4. Bayesian dropout [7].

Using the Bayesian dropout process, the predictions and confidence interval can be calculated as [7] as follow:

$$\mu_y = \frac{1}{N}\sum_{n=1}^{N} y_n \tag{11}$$

where μ_y is the mean of N predictions, and

$$\sigma_y^2 = \frac{1}{N-1}\sum_{n=1}^{N}(y_n - \mu_y)^2 \tag{12}$$

where σ_y is the variance. The range of the confidence interval provides information about any uncertainties related to the NN model used.

B. Gaussian Process (GP)

GP is the extension of standard multivariate Gaussian distribution to infinitely many variables, where any finite number of samples form a joint Gaussian distribution [7]. The prior of GP is defined by two quantities, namely a mean μ and a covariance matrix K, given as follow:

$$y = f(x) \sim \mathcal{N}(\mu(X), K_X) \tag{13}$$

where N represents a GP. From (13) the mapping between the input and output is enabled through the GP. For general nonlinear regression, a constant mean function $\mu(x) = m$ is used [13]. The kernel function $K(\boldsymbol{x})$ that describes the relation between points in the function is written as follow:

$$K(\boldsymbol{x}) = \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_1, \boldsymbol{x}_t) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_t, \boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_t, \boldsymbol{x}_t) \end{bmatrix}. \tag{14}$$

Appropriate kernel functions can be applied to capture different patterns in the dataset. For example, Matern kernels can be used when the function is less smooth. A commonly used kernel is the automatic relevance determination (ARD) [13] Matern 5/2 function given by [14]

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_f^2\left(1 + \sqrt{5}r + \frac{5}{3}r^2\right)e^{-\sqrt{5}r}$$
$$r = \left(\sum_{d=1}^{D}\frac{(\boldsymbol{x}_{i,d} - \boldsymbol{x}_{j,d})^2}{\sigma_d^2}\right)^{\frac{1}{2}} \tag{15}$$

where σ_f and σ_d are the hyperparameters of $K(\boldsymbol{x})$. These hyperparameters are updated during the training process by minimizing the negative log marginal likelihood of the GP to improve learning.

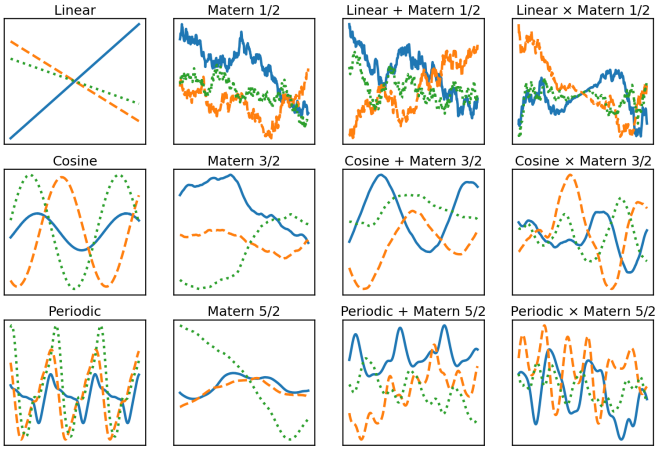


Fig. 5. Different GP kernels and their combinations.

We can also combine different standalone kernels to construct new kernels, which can then be applied to capture more complicated function behaviors, as illustrated in Fig. 5. Further details on kernels are provided in [15].

Once the GP model is trained using the dataset $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{Y}\}$, it can be used to predict the unknown response y^* for a new set of input data $x^* \in \mathbb{R}^{M\times D}$ using the relationship as follow:

$$p(y^*, Y|x^*, X, \theta) = \mathcal{N}\left(\begin{bmatrix}\mu X \\ \mu x^*\end{bmatrix}, \begin{bmatrix} K_X & K_{X,x^*} \\ K_{X,x^*}^T & K_{x^*,x^*}\end{bmatrix}\right) \tag{16}$$

where θ is the set of hyperparameters used as part of the training process.

During the training process, our goal is to find the best hyperparameters, θ, that fits the data and model. Fixing θ can create a combination of data and parameter related uncertainties due to any inaccuracies related to the model used for prediction. Therefore, we integrate over all possible θ and use a weighted sum of confidence intervals where the bounds obtained with multiple θ values affect the final confidence bound. This can be written as [16] as follow:

$$p(y^*|x^*, D_t) = \int p(y^*|x^*, D_t, \theta)p(\theta|D_t)d\theta \tag{17}$$

where $D_t = (X_t, Y_t)$ is the data at the tth iteration of the training process. At a test point x^*, the model predicts a distribution, $p(y^*|x^*, D_t)$, that no longer depends on θ and is a weighted sum of all possible distributions corresponding to fixed hyperparameter $p(y^*|x^*, D_t, \theta)$. We can use Markov chain MC (MCMC) [17] to learn the analytically intractable distribution $p(\theta|D_t)$. Once the GP is trained, the predictions and confidence intervals that also accounts for parameter-related uncertainties can be obtained as in [18].

C. Comparison of BNN and GP

To compare BNN and GP, we consider a differential plated through hole (PTH) pair in a package core along with the microvias in the build-up layers [6]. The 13-D input space and their corresponding range are provided in Fig. 6.

Here, the goal is to map 13 input parameters to their corresponding four-port S-parameters from DC to 100 GHz with

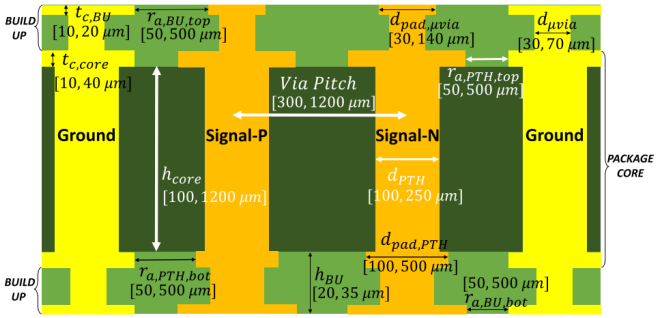


Fig. 6. Parameters of differential PTH in package core [16].

TABLE I
PTH COMPARISON

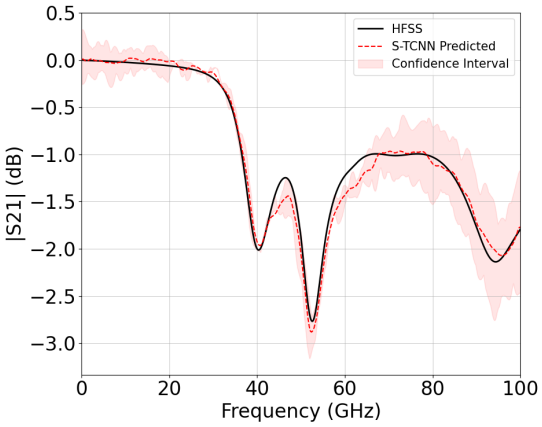
	Base S-TCNN	Bayesian S-TCNN	GP
Validation NMSE	0.033	0.037	0.034
Training Time	118.52s	121.72s	63.58s
Inference Time	-	0.91s	25.12s

100-MHz steps, corresponding to an output dimensionality of 12 000. To create the predictive model, 680 samples based on Latin hypercube sampling (LHS) are first determined. These are then fed into a full-wave EM solver to generate their corresponding S-parameters. After the data are collected, 550 out of 680 samples are used for the training of the model and the remaining are used for testing. The testing platform used is Intel Core i7-8750H, NVIDIA GeForce GTX 1070 Max-Q and 16GB RAM. All programming is performed with PyTorch [19]. The results are summarized in Table I.

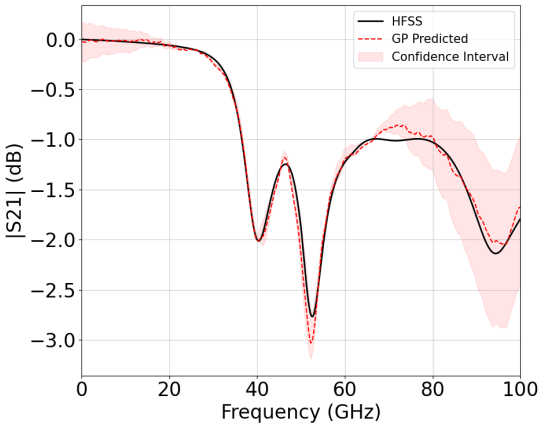
The predictions and uncertainty around the predictions of S21 for the PTH problem in Fig. 6 is shown in Fig. 7. As shown in Fig. 7(a) and (b), the BNN and GP provide uncertainties around the predictions, where the dashed line represents the mean. The BNN provides for a more general method as compared to GP which depends on the kernels used. Irrespective of the method used, both these techniques help establish confidence bounds around the predictions. Note that the time for prediction for the base and the Bayesian version is similar, but the Bayesian S-TCNN also provides confidence bounds in the same inference time. It is important to note from Fig. 7 that the EM simulator result [high-frequency structure simulator (HFSS)] in general falls within the mean and confidence bounds established by the NN.

III. DESIGN OPTIMIZATION

We next address optimization using Bayesian learning in two forms namely: 1) a generic Bayesian optimization (BO) method widely used in the ML community and 2) a customized optimization method specifically targeted for microwave structures. These are then compared with other ML and non-ML-based methods.



(a)



(b)

Fig. 7. Comparison of S21 predictions with confidence interval. (a) S-TCNN with Bayesian dropout. (b) GP.

A. Bayesian Optimization

The mathematical optimization problem can be written as follow:

$$\max_{\boldsymbol{x}} \; f(\boldsymbol{x}) \quad \text{s.t. } \boldsymbol{x} \in \mathcal{X} \tag{18}$$

where $\boldsymbol{x}$ is the vector of input parameters, $f(\boldsymbol{x})$ is the cost function, and $\mathcal{X}$ is the feasible range of input parameters, also called the design space. Linear and convex optimization problems (LP and CP) can be solved efficiently using iterative methods [20], [21] due to the properties of simple response surfaces. However, in several microwave design scenarios, the response surface is often nonconvex with many local minima and maxima, making optimization challenging. During the design of microwave structures, the function evaluation $f(\boldsymbol{x})$ often requires EM simulation which can take lengthy computations. In addition, due to the black-box nature of the problem, the gradient information is not available. Gradient-free global optimization algorithms have been proposed to solve optimization problems using heuristic methods [22]. For example, genetic algorithm (GA) has been used for 3-D integrated circuit design [23]. These algorithms mimic the process of natural selection and improves the performance

using mutation, crossover, and selection from a population of design solution candidates. This method has also been used to optimize the shape and topology of EM structures and has shown promising results [24]. Similarly, in particle swarm optimization (PSO) algorithm, a number of candidates, denoted as particles, are updated iteratively using the information of the global optimal particle position and the local optimal particle positions. The particles are guaranteed to converge to the global optima with sufficiently large number of particles and iterations [25]. Several applications of PSO in microwave design are discussed in [26], [27], and [28].

The majority of the microwave design problems require CPU-intensive simulations of multiscale and multiphysics structures, making the simulation time increase even more. BO is a well-known method for optimizing expensive black-box functions where a closed-form expression or surrogate model is unavailable [29], [30], [31]. The function to be optimized is treated as a GP where the posterior distribution of the function is obtained from the prior knowledge using the previously sampled points as mentioned in (13)–(15). This GP model is then used with an acquisition function $u(\boldsymbol{x})$ to determine the next evaluation point $\boldsymbol{x}_{t+1}$.

There are three commonly used acquisition functions, namely the probability of improvement (PI), expected improvement (EI), and upper confidence bound (UCB) given as follow:

$$Z = \frac{\mu(\boldsymbol{x}) - \widetilde{f}^* - \zeta}{\sigma(\boldsymbol{x})} \tag{19}$$

$$u_{\mathrm{PI}} = \Phi(Z) \tag{20}$$

$$u_{\mathrm{EI}} = \big(\mu(\boldsymbol{x}) - \widetilde{f}^* - \zeta\big)\Phi(Z) + \sigma(\boldsymbol{x})\phi(Z) \tag{21}$$

$$u_{\mathrm{UCB}} = \mu(\boldsymbol{x}) + K\sigma(\boldsymbol{x})$$

$$K = \sqrt{2\ln\big(2\pi M^2/(12\eta)\big)} \tag{22}$$

where $\widetilde{f}^*$ is the minimum cost evaluated so far, ζ is a hyperparameter, M is the number of function evaluations, $(1-\eta)$ is the probability of zero regret, and $\Phi(\cdot)$ and $\phi(\cdot)$ is the cumulative distribution function and probability density function of the normal distribution, respectively [30], [32]. The set of input parameters that maximizes $u(\boldsymbol{x})$ is selected as the next sampling points, $\boldsymbol{x}_{t+1}$. Since $u(\boldsymbol{x})$ is no longer a black-box function, the maximization procedure can be performed easily. The graphical illustration of BO is shown in Fig. 8. In summary, BO modifies the original optimization problem into a series of smaller and easier optimization problems in an alternate space to enable fast convergence to the optimum point in the original response surface.

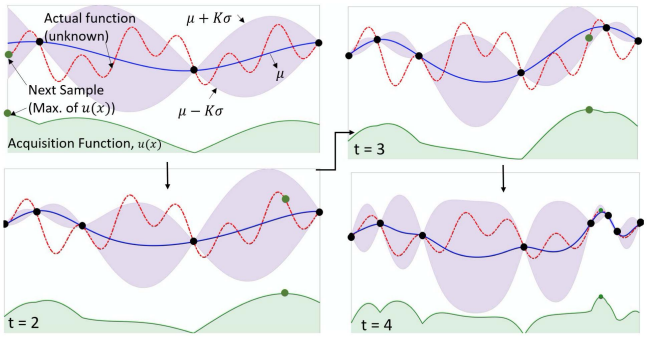


Fig. 8. Graphical illustration of BO [33].

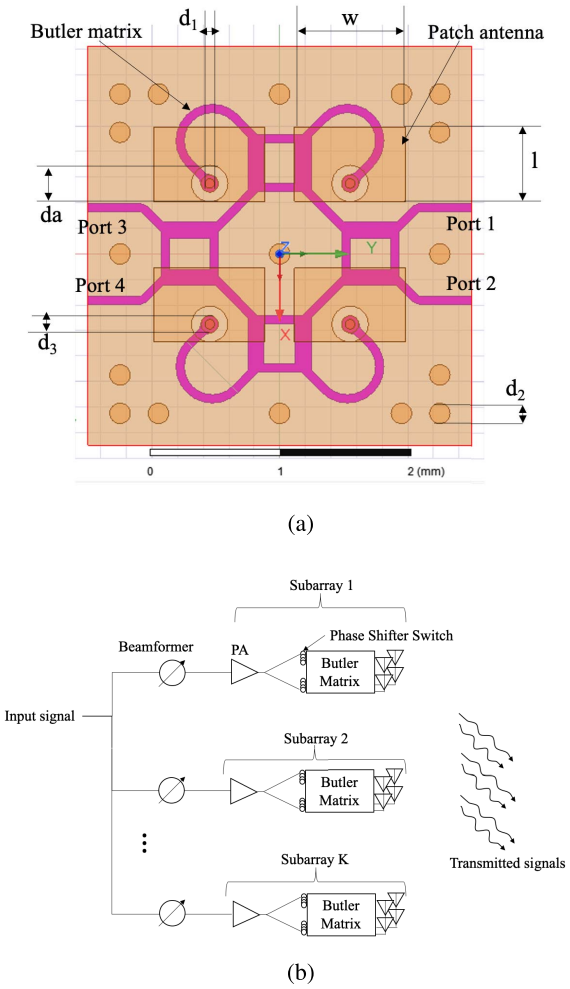


Fig. 9. Microstrip patch antenna beamformer array. (a) Geometrical configuration of subarray. (b) Signal flow.

B. BO Applied to Beamformer Design

Consider a patch antenna subarray design optimization problem. A Butler matrix subarray consisting of four microstrip antennas is shown in Fig. 9(a) [34]. The Butler matrix subarray is designed such that every four elements have eight phase shift combination options controlled by the phase shifter switch. These switches are implemented by sending power to the corresponding ports. Each phase shift combination creates a beam pointing along a certain direction. Each subarray along with a power amplifier (PA) is connected to a beamformer chip, as shown in Fig. 9(b). The PAs are used to amplify signals while maximizing the power efficiency. The beamformers provide continuous phase shifts which are tunable according to the desired beam direction. Several subarrays as in Fig. 9(b) are connected together to steer the beam along a 2-D direction.

The optimization problem can be defined as follow:

$$\min_{\boldsymbol{p}} \int_{\Theta_d} J(\theta_d; \boldsymbol{p}) d\theta_d \tag{23}$$

where $\boldsymbol{p} \in R^L$ is the geometrical configuration parameter vector. In this example, $L = 7$ represents the number of geometrical configuration parameters, $\theta_d \in R^2$ is the desired beam direction in spherical coordinates, and Θ_d is the set of possible desired beam directions. In (23), $J(\theta_d)$ is the cost function that captures the performance degradation due to the radiation toward undesired angles when the desired beam direction is θ_d, which needs to be minimized. $J(\theta_d; \boldsymbol{p})$ is defined as follow:

$$J(\theta_d, \boldsymbol{\xi}, \boldsymbol{\psi}; \boldsymbol{p}) = \frac{\max_{\theta_c \in \Theta_c} |G(\theta_c, \boldsymbol{\xi}, \boldsymbol{\psi}; \boldsymbol{p})|^2}{|G(\theta_d, \boldsymbol{\xi}, \boldsymbol{\psi}; \boldsymbol{p})|^2} \tag{24}$$

where $\boldsymbol{\xi} \in C^K$ is the beamformer vector, K is the number of beamformers (same as the number of subarrays), $\boldsymbol{\psi}$ indicates the positions of the phase shifter switches, $\theta_c \in R^2$ is the interference direction in spherical coordinates, Θ_c is the set of interfering directions given θ_d, and $G(\theta, \boldsymbol{\xi}, \boldsymbol{\psi}; \boldsymbol{p})$ is the complex response of the antenna along the direction θ using beamformers $\boldsymbol{\xi}$ and phase shifter switches $\boldsymbol{\psi}$ given as follow:

$$G(\theta; \boldsymbol{\xi}, \boldsymbol{\psi}, \boldsymbol{p}) = \sum_{k=1}^{K} g_{\text{sub},k}(\theta, \boldsymbol{\psi}_k, \boldsymbol{p}) e^{-2\pi i f \tau_{\text{sub},k}(\theta)} \xi_k \tag{25}$$

$$\tau_{\text{sub},k}(\theta) = -\frac{\langle \boldsymbol{x}_{\text{sub},k}, \boldsymbol{d}(\theta) \rangle}{c} \tag{26}$$

where $g_{\text{sub},k}(\theta, \boldsymbol{\psi}_k, \boldsymbol{p})$ is the radiation pattern of the kth subarray (computed using EM simulators such as Ansys HFSS) given the subarray structure in (25). In (25), $\boldsymbol{\psi}_k$ is the phase shifter switch of the kth subarray, $\tau_{\text{sub},k}(\theta)$ is the time delay between the signal transmitted from the kth subarray and the center of the entire array, $\boldsymbol{x}_{\text{sub},k}$ is the position of the center of the kth subarray, $\boldsymbol{d}(\theta)$ is the unit vector of the angle direction θ transformed to the Cartesian coordinate, and c represents the speed of light and $\langle \cdot, \cdot \rangle$ is the standard inner product in $\mathbb{R}^3$.

We repose (23) for numerical computations in the form as follow:

$$\min_{\boldsymbol{p}} \frac{1}{N} \sum_{\theta_d \in \Theta_d} J(\theta_d; \boldsymbol{p}) \tag{27}$$

where N is the number of desired beam angle samples in Θ_d. In this example, we apply the same phase shifter switches to all Butler matrices, and set the switches to maximize the Butler matrix response along the desired beam direction. In addition, the beamformers are assumed to compensate for the delays due to the position difference of the subarrays. In other words

$$\begin{aligned} \boldsymbol{\psi}_1(\theta_d) &= \boldsymbol{\psi}_2(\theta_d) \\ &= \cdots = \boldsymbol{\psi}_K(\theta_d) \\ &= \boldsymbol{\psi}^*_{\text{sub}}(\theta_d) \end{aligned} \tag{28}$$

$$\boldsymbol{\psi}^*_{\text{sub}}(\theta_d) = \arg\max_{\boldsymbol{\psi}_{\text{sub}}} \left| g_{\text{sub}}(\theta_d, \boldsymbol{\psi}_{\text{sub}}, \boldsymbol{p}) \right|^2 \tag{29}$$

$$\xi_k(\theta_d) = e^{2\pi i f \tau_{\text{sub},k}(\theta_d)}. \tag{30}$$

In our experiment, K is set to 16 corresponding to the 4 by 4 subarrays, $N = 19$ indicating 5° increment sampling between 0°∼90° azimuth while keeping the elevation as 30°, where UCB is chosen as the acquisition function and the iteration count is 50. By simply applying BO, the optimized parameters reduce the cost function significantly. Since the result of BO depends on the predetermined bound of the search space, we further modify BO by adopting an adaptive bound where the bound gradually shrinks as the iteration increases centering at the latest optima. The adaptive bound can be defined as follow:

$$\left[\boldsymbol{p}_t - \frac{1}{2\exp(t/\alpha)}, \boldsymbol{p}_t + \frac{1}{2\exp(t/\alpha)} \right] \tag{31}$$

where $\boldsymbol{p}_t$ is the optima obtained so far within t iterations, and α is the hyperparameter representing the negative shrinking rate of the bound. This helps the algorithm focus on a smaller region, and center the region around the current optima. The cost [based on (27)] comparison between the manual design [34], the design using PSO, the conventional BO, and the modified BO is shown in Fig. 10 indicating the advantage obtained using the modified BO. Fig. 11 shows the comparison between the beam patterns from the optimization methods described where the mainbeam is pointing toward $\theta_d = [30, 90]$. The radiation power is normalized with respect to the mainbeam of the original design. The result shows that the power of the mainbeam is increased and the sidelobe is reduced after optimization. The comparison between the optimized design and the original hand-tuned design from [34] is shown in Table II.

TABLE II
DESIGN PARAMETERS FOR BEAMFORMER ARRAY

Parameters		Values	
		Hand-tuned	Optimized
Patch width	w	870 μm	753.4 μm
Patch length	l	580 μm	538.6 μm
Cu thickness	t	20 μm	20.2 μm
Via diameter	d_1	75 μm	81.6 μm
Via diameter	d_2	160 μm	109.6 μm
Cu-plate hole diameter	da	285 μm	213.8 μm
Pad diameter	d_3	135 μ m	138.8 μm

C. Deep-Partitioning Tree BO

In the previous section, a typical BO approach was illustrated where the optimization is done on the acquisition function. When the dimensionality increases manifold, this optimization process can become computationally expensive by itself. In addition, a specific acquisition function (EI) was selected, that may not be the appropriate one if the design problem changes. In this section, a high-dimensional BO method is illustrated, called BO with deep partitioning tree (DPT-BO) [35], which uses an additive GP (ADD-MES-G) to approximate a high-dimensional objective function. In addition, the additive structure used preserves the interaction between parameters to capture various classes of design problems that can be addressed. This makes the DPT-BO method

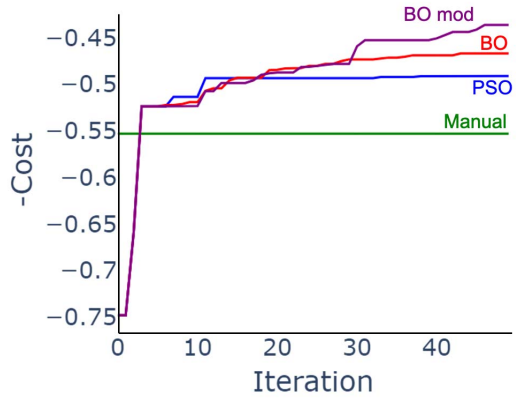


Fig. 10. Cost comparison between the manual design, the design from PSO, the conventional BO, and modified BO. Manual here refers to the design process without the use of optimization.

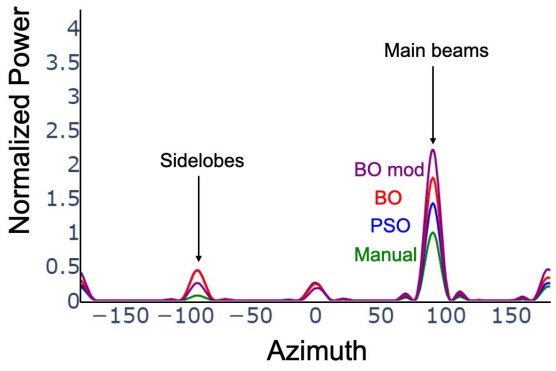


Fig. 11. Antenna beam pattern comparison.

particularly applicable to high-frequency electronic design problems.

Preserving such interactions makes the auxiliary optimization of acquisition function in BO very challenging. Therefore, a deep partitioning tree method is used, that completely eliminates the optimization of the auxiliary function and uses instead sensitivity of input parameters to determine where to query the function next. The sensitivities are learned by utilizing ARD kernels for the GP as described in [35].

As an example, we present an inductive coupling-based wireless power transfer (WPT) system operating at 1 GHz that consists of a layout and circuit components. The architecture of the WPT system is shown in Fig. 12 consisting of embedded rectangular RF coils connected to transmit (TX) and receive (RX) matching networks, a full-bridge diode rectifier, and a buck converter (BC) for DC regulation.

The control parameters for the WPT system are shown in Table III. The multiobjective optimization problem can be posed as follow:

$$f(x) = \Sigma_{i=1}^{3} w_i y_i \tag{32}$$

where y_1, y_2, and y_3 are the rectifier efficiency ($P_{\text{out}}/P_{\text{in}}$), minimum input voltage of the BC, and area of RX coil with $w_1 = 7$, $w_2 = -3.5$, and $w_3 = -3$ being the corresponding weights, chosen to prioritize efficiency over area. The data

TABLE III
DESIGN PARAMETERS FOR WPT SYSTEM (FIG. 12)

Parameter		Unit	Min	Max
Inner Height of TX coil	$g_{y,TX}$	mm	1	5
Inner Height of RX coil	$g_{y,RX}$	mm	1	5
Inner Width of TX coil	$g_{x,TX}$	mm	1	5
Inner Width of RX coil	$g_{x,RX}$	mm	1	5
Line Width of TX coil	$l_{w,TX}$	mm	0.5	3
Line Width of RX coil	$l_{w,RX}$	mm	0.5	3
Feeding Gap for TX coil	$g_{f,TX}$	mm	0.5	3
Inner Height of RX coil	$g_{f,RX}$	mm	0.5	3
TX Vertical GND Cut-out Ratio	$slot_{y,TX}$		0.8	1.2
RX Vertical GND Cut-out Ratio	$slot_{y,RX}$		0.8	1.2
TX Horizontal GND Cut-out Ratio	$slot_{x,TX}$		0.8	1.2
RX Horizontal GND Cut-out Ratio	$slot_{x,RX}$		0.8	1.2
Capacitor I	C_1	pF	0.1	10
Capacitor II	C_2	pF	0.1	10
Capacitor III	C_3	pF	0.1	10
Capacitor IV	C_4	pF	0.1	10
Inductor I	L_1	nH	0.1	10
Inductor II	L_2	nH	0.1	10
Input Power	$P_{RF,IN}$	dBm	5	15
Widths of all TLINs	$w_{TL1},\ldots,_{TL13}$	mil	15	45

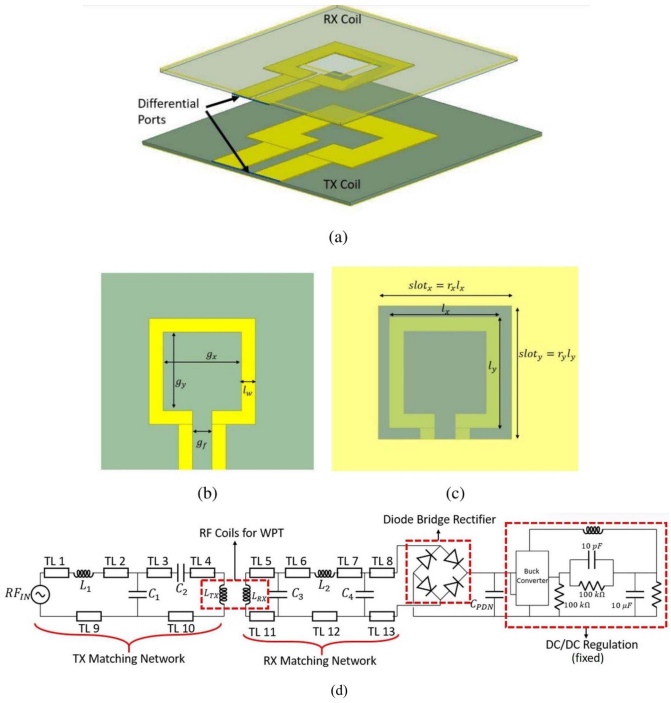


Fig. 12. (a)–(c) Geometry of the embedded RF coils defined by the control parameters. (d) Schematic of the WPT-based power delivery architecture [36].

samples are chosen from the output of a EM simulator (Ansys HFSS). The convergence of DPT-BO is compared with ADD-MES-G (another generic BO method), and PSO along with the final results in Table IV. From Fig. 13, both DPT-BO and ADD-MES-G have a sharp convergence over the first few iterations which is typical of BO type algorithms as compared to PSO. Due to the customization of the DPT-BO algorithm to such type of RF problems, the system efficiency

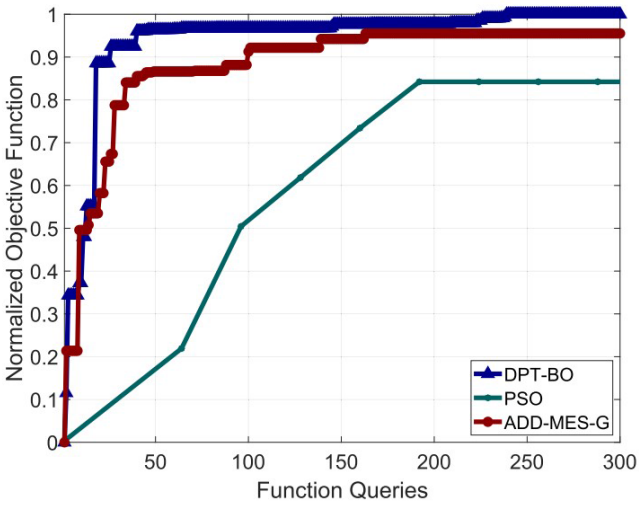


Fig. 13. Performance comparison of the proposed algorithm on maximizing objective function in (32).

TABLE IV
OPTIMIZATION RESULTS FOR WPT SYSTEM

	PSO	ADD-MES-G	DPT-BO
RX Coil Area (mm^2)	7.48	19.26	11.04
Rectifier Efficiency	53.25	65.72	66.96
Input Voltage of BC (V)	3.24	2.61	2.72
System Efficiency	45.83	58.86	59.57
AUC (Normalized)	1.50	1.07	1.00

achieved though similar to ADD-MES-G, results in smaller RX coil area. In comparison, though PSO results in smaller RX area, the resulting efficiencies are much smaller than the BO methods. The normalized AUC values show that DPT-BO converged 1.50 × and 1.07 × faster than PSO and ADD-MES-G, respectively.

IV. INVERSE DESIGN

The aforementioned ML techniques learn the relationships between the input and output parameters through a model, where the output response is predicted given the input parameters. Inverse design represents the process of estimating the input parameters based on a set of desired output parameters. In this section, we introduce inverse design techniques and apply them to microwave design.

A. Techniques for Inverse Design

Inverse problems are inherently ill-posed and intractable. The problem of invertibility poses three questions. 1) Does the inverse exist? 2) If the inverse exists, is it unique? and 3) Is the inverse stable? Several architectures have been proposed to address the problem of invertibility. Inverse methods have been introduced using space-mapping (SM) [37], where, rather than apply optimization directly to an expensive high fidelity or fine model, a low fidelity or coarse model is used to achieve the same results using information from the coarse model. A linear inverse SM (LISM) optimization algorithm is described in [38] for designing microwave circuits in the frequency- or time-domain transient state. LISM provides a simplified implementation with respect to the neural inverse SM (NISM) by approximating the inverse of the mapping function at each iteration. It continues to state that LISM follows an aggressive formulation in the sense of not requiring up-front fine model evaluations. This method has been applied to CMOS drivers to find the optimal channel widths that result in the desired output voltage specifications. In [39], Simsek and Sengor propose a method based on SM with inverse difference (SM-ID) to solve inverse problems. They modify the SM with difference (SM-D) method using a well-known feedforward NN structure to obtain an inverse coarse model, and refine the space mapping function by building a mapping between the inverse coarse model output, the inverse coarse model design parameters, and fine model design parameters. This has been applied to the shape reconstruction of a conducting cylinder.

Physics-based surrogates exhibit excellent generalization but require an underlying low-fidelity model which is not always available or sufficiently reliable [40]. In [41], a ML surrogate model is introduced for the inverse problem that relies on the least-squares support vector machine (LS-SVM) to provide an accurate relationship among the desired eye features and the geometrical parameters of a high speed link. In [40], a metamodel-based procedure for design closure acceleration is presented, involving two kriging interpolation surrogates: an inverse model that provides a good initial point for subsequent optimization, which is further augmented by the Jacobian matrix estimated using the forward model. Pietrenko-Dabrowska *et al.* [40] implement this procedure using a compact three-section impedance matching transformer and report that their approach permits reliable optimization at a low cost of a few EM simulations of the structure at hand.

In recent years, ANNs have been deployed as an effective tool for microwave design and modeling problems [42]. The NN models that have been proposed to solve the inverse problem fall largely into two categories: 1) evaluating models iteratively to find the optimal solutions for the specified output response and 2) training the input and output nodes by transposing them [42]. In [43], the applicability of NNs in search of a design solution is proposed by implementing an optimization routine through a learning process. This method aims to convert conventional circuit models into NN models where the method is demonstrated with the design of heterojunction bipolar transistor amplifiers with 11 parameters with the frequency response as the output specification. In [42], the problem of non-uniqueness in inverse modeling is addressed through multivalued solutions using adjoint NN derivative information to separate training data into groups. This inverse modeling methodology has been applied to waveguide filter design and validated by comparing the NN solutions with measurements from the filters. In [44], a lifelong learning architecture is proposed using deep learning where multiple predictions and classifications are done jointly and applied for inverse mapping of transmission line geometries based on eye characteristics desired. To address the non-uniqueness

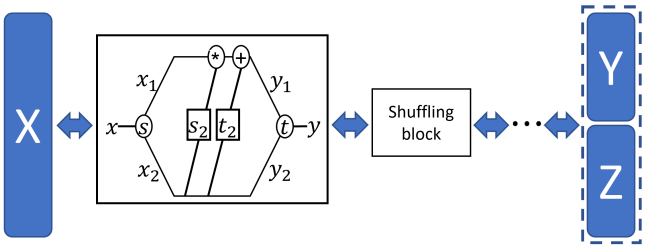


Fig. 14. Architecture of the invertible NN (INN) (X: input, Y: output, Z: latent variable) [47].

problem of inverse design, a multivalued NN inverse modeling method is proposed in [45] where the model learns multiple solutions of geometrical parameters at the output from the electrical response at the input. The method in [45] uses a training error function to associate the output–input tuples in a unique fashion, which is demonstrated on microwave filters.

In addition, the state-of-the-art generative models have been developed to output posterior conditional distributions instead of a deterministic design solution. In this section, we provide cursory look at how we leverage the use of Bayes' update in generative modeling for inverse design.

In generative modeling, we utilize the full Bayesian approach described in (7) by marginalizing (summing) over parameter θ. Given input x and output y, the full Bayesian approach is expressed as [46] as follow:

$$\begin{aligned} p(y|x) &= \frac{p(y,x)}{p(x)} \\ &= \frac{\int_\theta p(y,x,\theta)\,d\theta}{p(x)} \\ &= \frac{\int_\theta p(y|x,\theta)p(\theta|x)p(x)\,d\theta}{p(x)} \end{aligned} \tag{33}$$

using the chain rule. Here, (y, x) represents the output-input data tuple, and θ represents the set of parameter defining the relationship between x and y. This leads to [46]

$$p(y|x) = \int_\theta p(y|x,\theta)p(\theta|x)\,d\theta. \tag{34}$$

We adapt some novel approaches in generative modeling from Bayesian learning, and redefine them here for microwave design.

B. Invertible NN (INN)

The questions bordering on existence, uniqueness, and stability of inverse solutions can be addressed in flow-based generative models, such as the INN [48], [49]. Given a sample x from design space X and its probability density $p_X(x)$, the corresponding y from the response space Y and its unknown probability density $p_Y(y)$ related through the transformation $Y = f(X)$, we can form a relationship between their probability densities through the change-of-variables technique [48], [50] as follow:

$$p_Y(y|\theta) = p_X\left(f_\theta^{-1}(y)\right) \cdot \left|\det\left(\frac{df_\theta^{-1}}{dx}\right)\right| \tag{35}$$

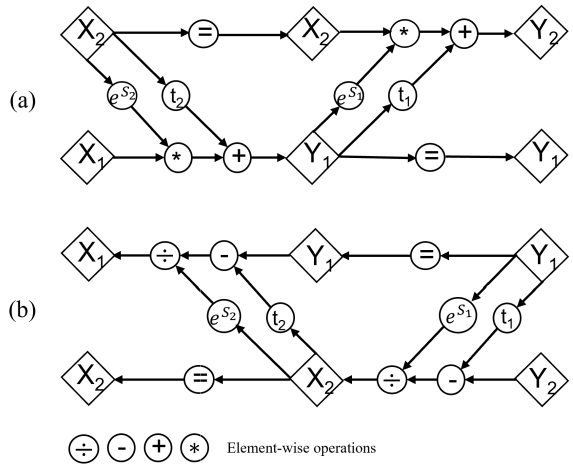


Fig. 15. Computational graphs for (a) forward and (b) inverse propagation [48].

where we define all the composition of the INN architecture in a single function f_θ, where θ is the set of all network parameters. The INNs are made of reversible blocks, and they can be trained in both directions simultaneously, as shown in Fig. 14. In addition to the outputs y of the system, a set of latent variables z can be defined which encode the lost information in the forward direction. Variables z can be sampled from a standard normal distribution, which, when passed through the trained network in the reverse direction, conditioned on an output y, result in the conditional posterior distributions $p(x|y)$. The INN is made up of stacks of affine coupling blocks as shown in Fig. 15. The block's input vector is halved into $[x_1, x_2]$, and they are transformed by an affine function with coefficients e^s and t, given by [48], [49]

$$\begin{aligned} y_1 &= x_1 \\ y_2 &= x_2 \circ e^{s(x_1)} + t(x_1). \end{aligned} \tag{36}$$

Given the block's output $[y_1, y_2]$, these expressions are invertible through [48], [49]

$$\begin{aligned} x_1 &= y_1 \\ x_2 &= (y_2 - t(y_1)) \circ e^{-s(y_1)} \end{aligned} \tag{37}$$

where (36) represents the forward mapping while (37) represents the inverse mapping (see Fig. 15 for a graphical illustration). The use of element-wise additive (+) and multiplicative ($\circ$) operations allows the inverse of the transformation to be easily computed without requiring the scale $s(\cdot)$ and shift $t(\cdot)$ networks to be inverted. The bijectivity of the INN model allows for bidirectional operation and training, and therefore both forward and inverse processes can be well learned [51]. The losses in the forward direction are: 1) supervised loss, which is the mean square error (MSE) between the true observations and predicted values; 2) unsupervised loss on the joint distributions of the network outputs and the product of marginal distributions of the simulation outputs and known latent distributions; and 3) unsupervised loss on the distribution of the backward predictions and known prior distribution.

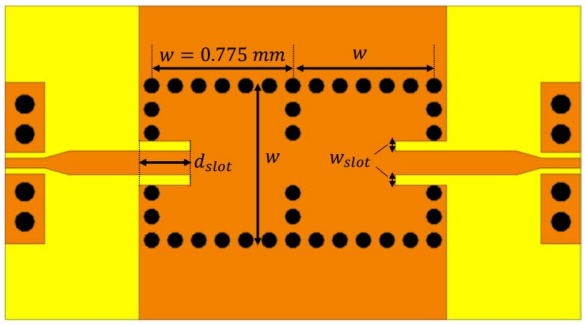


Fig. 16. Structure of a second-order SIW filter [52].

TABLE V
SIW FILTER DESIGN SPACE [52]

Parameter		Unit	Min	Max
Slot width	w_{slot}	μm	40	120
Slot Depth	d_{slot}	μm	150	300

TABLE VI
PERFORMANCE OF INVERSE DESIGN CANDIDATES [52]

	Parameter		Performance	
	w_{slot} μm	d_{slot} μm	f_c GHz	Max dB/GHz
Design target	-	-	142	2.6
Candidate 1	51	261	141.4	2.61
Candidate 2	106	176	142.6	2.93

TABLE VII
DESIGN PARAMETERS OF GILBERT CELL MIXER

Parameter		Unit	Min	Max	Step
Collector resistor	R_c	Ω	50	450	50
Tail resistor	R_t	Ω	20	220	50
Base resistor (top)	R_{b1}	Ω	10	160	30
Base resistor (bottom)	R_{b2}	Ω	10	90	20
Coupling capacitor	C_c	μF	0.4	1.2	0.2
Bypass capacitor	C_b	μF	0.4	1.2	0.2

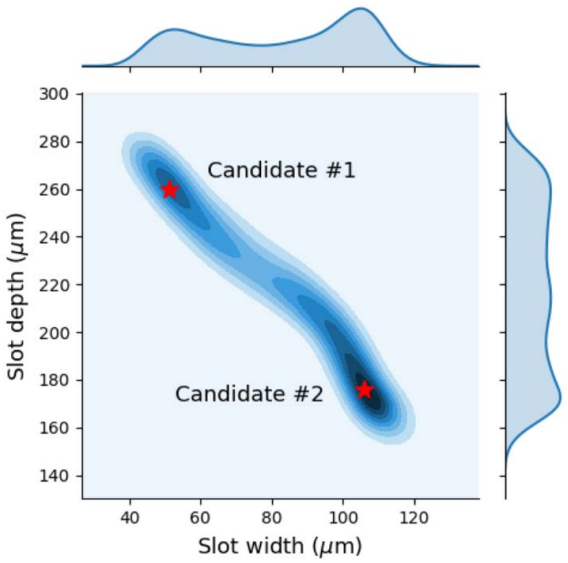


Fig. 17. Predicted conditional posterior distribution of the design parameters from the trained INN model. Candidate points are marked as red stars [52].

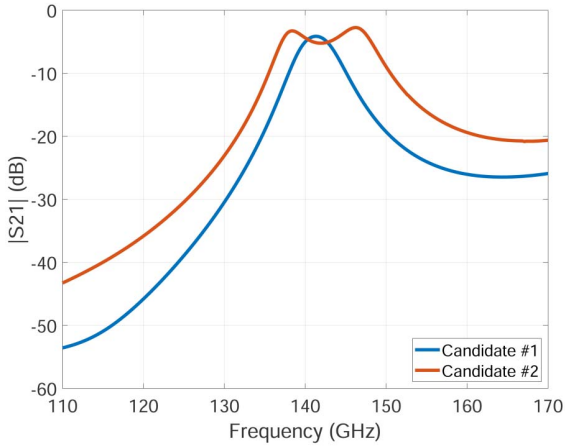


Fig. 18. Insertion loss for the two candidate designs obtained with INN [52].

C. Example 1—Inverse Design of SIW Filters in D-Band

We apply the INN method to a substrate-integrated waveguide (SIW) filter used in D-band. Here, the geometry of the structure is shown in Fig. 16, which constitutes a second-order SIW filter with polymer-coated glass substrate and has a total thickness of 130 μm. Permittivity and loss tangent of the polymer and the glass material are $\epsilon_r = (3.2, 4.9)$ and $\tan\delta = (0.044, 0.0056)$, respectively. The width of the filter is 755 μm. The input design space parameters are the slot width w_{slot} and slot depth d_{slot} of the feeding structure as shown in Table V. The output specifications are the center frequency f_c of the passband, and the roll-off which is the slope of S_{21} in dB at lower cut-off frequency where

$$\text{roll-off} = \frac{\partial S_{21}(\text{dB})}{\partial \text{frequency(GHz)}}. \tag{38}$$

We construct an INN model using eight reversible blocks with permutation layers between them. In each reversible block, the scale $s(\cdot)$ and shift $t(\cdot)$ networks are made of fully connected NNs with one hidden layer of 54 neurons and rectified linear unit (ReLU) activation functions. For better modeling capability, the inputs (w_{slot} and d_{slot}) are zero-padded to 16 dimensions. The output variables consist of the target characteristics (center frequency and roll-off) and 2-D latent variables z sampled from a standard normal distribution, and are zero-padded to 16 dimensions as well. To create the training data, we randomly generate 150 samples from the uniform distributions of the input parameters and feed into an EM solver, Ansys HFSS, for the simulation of the SIW filter [51]. The INN training takes a few seconds for the 150 samples. For the training samples, f_c falls into the range of 136.8–163.1 GHz, and roll-off falls into the range of 1.51–5.28 dB/GHz.

The trained model generates joint posterior distributions of the input design space parameters, as shown in Fig. 17, for a desired response of $f_c = 142$ GHz and roll-off $=$ 2.6 dB/GHz. We identify multiple candidate design regions.

TABLE VIII
BAYESIAN LEARNING TABLE

Tasks	Base model	Bayesian Technique	Application
Optimization	PSO, DPT-BO	GP, acquisition function	Buttler matrix [34], Integrated system design [32], Wireless power transfer [35]
Uncertainty Quantification	STCNN	GP, Dropout	PTH [4]
Inverse Design	INN	Jacobian Update, Normalizing flows	High-speed channel design [58], Substrate Integrated Waveguide (SIW) filter [51] Active mixer

To verify our trained model, we pick two candidate designs that correspond to the peak distribution density points in the two regions, marked as red stars in Fig. 17. We simulate the candidate points in Ansys HFSS and display their responses in Table VI, while their S_{21} plots are shown in Fig. 18. We find that design candidate two shows typical characteristics of a second-order filter, a two-notch passband with the desired cut-off frequency and roll-off performance. As for design candidate one, it shows a significantly higher Q-factor and roll-off performance without any ripples in the passband. These inverse solutions demonstrate the validity of INN. The two designs give desired performance: 1) an intuitive solution with the expected characteristics and impedance matching and 2) an unintuitive design with a significantly higher Q-factor that is obtained by changing the input impedance of the SIW.

D. Example 2—Inverse Design of Active Mixer

We further validate the INN method with an inverse design of active mixer, which is a nonlinear device used for summing and subtracting frequencies. They are characterized by their conversion gain or loss and how much noise they introduce in the circuit. Consequently, accurate nonlinear modeling of mixers are crucial to getting good performance. Consider the IAM-81018 Gilbert cell mixer [53], [54], [55], shown in Fig. 19. The mixer is a down converter, with an RF of 2 GHz and a 250 MHz IF, operating from a 5 V DC supply. The objective here is to obtain the mixer design parameters that satisfy a given specification of gain and noise figure. The design parameters of the mixer are the passive components as shown in Fig. 19, and their range of values are given in Table VII. The target characteristics investigated are the conversion gain G and the noise figure NF, given as [55], [56], [57] as follow:

$$G\ (\text{dB}) = P_{\text{IF}} - P_{\text{RF}} \tag{39}$$

and

$$\text{NF} = \frac{kTBG + N_{0(\text{mixer})}}{kTBG} \tag{40}$$

where P_{IF} and P_{RF} are the powers at the IF and RF ports, respectively, $kTBG$ and $N_{0(\text{mixer})}$ are the source noise and noise added by the mixer (both referred to the IF port), respectively.

We generate 27 000 training samples in a uniform fashion using keysight ADS [55], and obtain the gain and noise figure

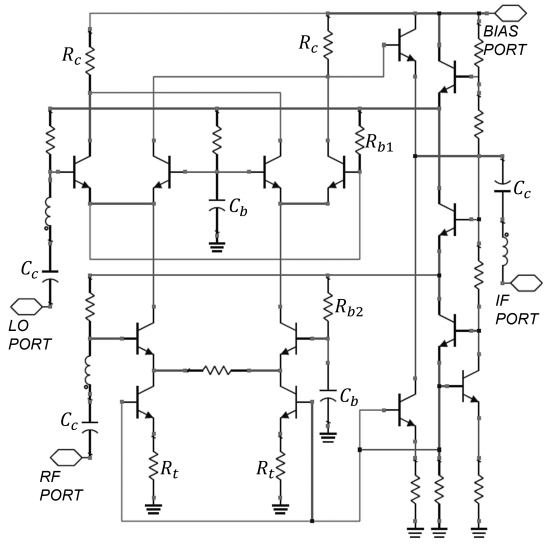


Fig. 19. IAM-81018 Gilbert cell mixer schematic [53], [54], [55].

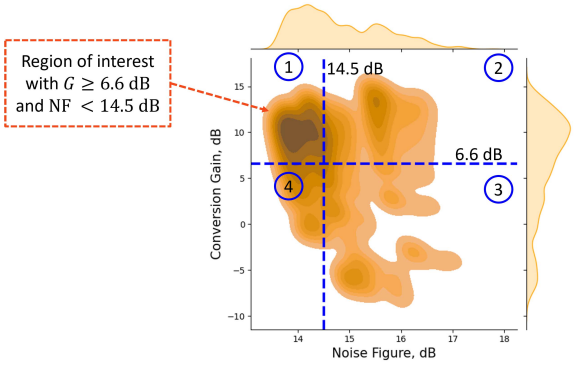


Fig. 20. Joint distribution of the conversion gain G and noise figure NF. It is partitioned into a grid of 4 cells based on the target specification $G \geq 6.6$ dB and NF < 14.5 dB.

for the corresponding mixer configuration. Fig. 20 shows the joint distribution of the gain and noise figure. The goal of the mixer design is to meet an arbitrary specification of a minimum gain of 6.6 dB and a maximum noise figure of 14.5 dB. Based on this desired target, the output space of the joint distribution of the gain and noise figure can be partitioned into a grid of smaller cells (four in total), as shown in Fig. 20. Using one-hot vector, output y can be represented using one of

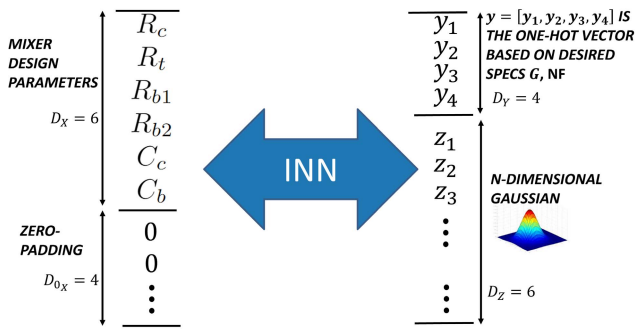


Fig. 21. INN model setup for mixer design.

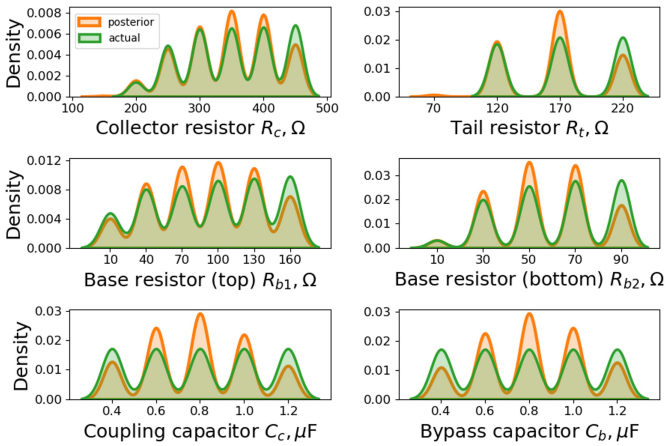


Fig. 22. Predicted conditional posterior distributions $p(x|y_{\text{target}} = \{G = 6.6\text{ dB}, \text{NF} = 14.5\text{ dB}\})$ of mixer design parameters, when passed through a smoothing window such as the kernel density estimator [46]. Light green indicates the actual distributions, light orange indicates the generated INN distributions, and tan indicates the overlap between the actual and INN distributions.

four values of a four-bit one-hot vector from the set as follow:

$$y = \begin{cases} \{1000\}, & G \geq 6.6\text{ dB and NF} < 14.5\text{ dB} \\ \{0100\}, & G \geq 6.6\text{ dB and NF} \geq 14.5\text{ dB} \\ \{0010\}, & G < 6.6\text{ dB and NF} \geq 14.5\text{ dB} \\ \{0001\}, & G < 6.6\text{ dB and NF} < 14.5\text{ dB}. \end{cases} \tag{41}$$

In this example, the proposed model setup is shown in Fig. 21. The INN model is constructed using eight reversible blocks with shuffling layers between them. In each reversible block, the scale $s(\cdot)$ and shift $t(\cdot)$ networks are made of fully connected NNs with one hidden layer of 64 neurons and ReLU activation functions. On the input side of the model setup, there are 6 mixer design parameters, zero-padded to ten dimensions. The output variables consist of the four-bit one-hot vector [as described in (41)] based on the desired mixer specifications (G and NF) and 6-D latent variables z sampled from a standard normal distribution, with no zero-padding. With this model setup, we train the INN for 300 epochs with 10 iterations per epoch. During the inference process, we generate rich conditional posterior distributions of the mixer design parameters as shown in Fig. 22, which gives the designer latitude when sampling from the solutions and also complying with the process rules and other constraints. We overlay the INN distributions with the actual distributions for comparison in Fig. 22. From the figure, we find that there is good correlation between the actual distributions and the distributions generated by INN, with the INN revealing some non-intuitive regions of higher densities.

V. Conclusion

As shown through the examples related to microwave design and analysis, the application of Bayes theorem combined with ML referred to here as Bayesian learning, provides advantages and opportunities for both forward and inverse modeling. Since quantifying uncertainties, high-dimensional optimization, and inverse problem modeling play an important role in microwave design, we believe that Bayesian learning has an important role to play in these areas. We summarize on a task basis, Bayesian learning scenarios in Table VIII. We enlist the base models as well as the technique used for approximate or exact Bayesian inference. We also mention the different applications where such approaches have been used. Bayesian learning involves updating the belief based on observed data, which when combined with ML can provide a powerful framework for microwave design and optimization.

Besides the illustration of various Bayesian learning scenarios in this work, the authors feel that there is still a long way to go for improvements in Bayesian learning in the context of microwave devices, components and integrated systems. This article highlights some initial work in this area.

References

[1] S. B. McGrayne, *The Theory That Would Not Die: How Bayes' Rule Cracked the Enigma Code, Hunted Down Russian Submarines, and Emerged Triumphant From Two Centuries of Controversy*. New Haven, CT, USA: Yale Univ. Press, 2011.

[2] *Bayesian Analysis #1: Concepts*. Accessed: Mar. 16, 2022. [Online]. Available: https://kevintshoemaker.github.io/NRES-746/LECTURE6.html

[3] J. Cui, F. Feng, W. Na, and Q.-J. Zhang, "Bayesian-based automated model generation method for neural network modeling of microwave components," *IEEE Microw. Wireless Compon. Lett.*, vol. 31, no. 11, pp. 1179–1182, Nov. 2021.

[4] M. Swaminathan, H. M. Torun, H. Yu, J. A. Hejase, and W. D. Becker, "Demystifying machine learning for signal and power integrity problems in packaging," *IEEE Trans. Compon., Package., Manuf. Technol.*, vol. 10, no. 8, pp. 1276–1295, Aug. 2020.

[5] H. M. Torun *et al.*, "A spectral convolutional net for co-optimization of integrated voltage regulators and embedded inductors," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Nov. 2019, pp. 1–8.

[6] H. M. Torun, A. C. Durgun, K. Aygün, and M. Swaminathan, "Enforcing causality and passivity of neural network models of broadband S-parameters," in *Proc. IEEE 28th Conf. Electr. Perform. Electron. Packag. Syst. (EPEPS)*, Oct. 2019, pp. 1–3.

[7] H. M. Torun, "Machine learning based design and optimization for high-performance semiconductor packaging and systems," Ph.D. dissertation, Georgia Inst. Technol., Atlanta, GA, USA, 2020.

[8] J. R. Hershey and P. A. Olsen, "Approximating the Kullback Leibler divergence between Gaussian mixture models," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2007, pp. IV-317–IV-320.

[9] Z. Ghahramani and M. Beal, "Variational inference for Bayesian mixtures of factor analysers," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 12, 1999, pp. 1–7.

[10] C. Rasmussen and Z. Ghahramani, "Occam's razor," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 13, 2000, pp. 1–7.

[11] Y. Gal *et al.*, "Uncertainty in deep learning," Ph.D. thesis, Univ. Cambridge, 2016.

[12] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1050–1059.

[13] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian optimization of machine learning algorithms," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 25, 2012, pp. 1–4.

[14] C. K. Williams and C. E. Rasmussen, *Gaussian Processes for Machine Learning*, vol. 2. Cambridge, MA, USA: MIT Press, 2006.

[15] D. Duvenaud, "Automatic model construction with Gaussian processes," Ph.D. dissertation, Dept. Comput. Sci., Univ. Cambridge, Cambridge, U.K., 2014.

[16] M. Swaminathan, H. M. Torun, H. Yu, J. A. Hejase, and W. D. Becker, "Demystifying machine learning for signal and power integrity problems in packaging," *IEEE Trans. Compon., Package., Manuf. Technol.*, vol. 10, no. 8, pp. 1276–1295, Aug. 2020.

[17] R. M. Neal, "Slice sampling," *Ann. Statist.*, vol. 31, no. 3, pp. 705–767, Jun. 2003.

[18] H. M. Torun, J. A. Hejase, J. Tang, W. D. Beckert, and M. Swaminathan, "Bayesian active learning for uncertainty quantification of high speed channel signaling," in *Proc. IEEE 27th Conf. Electr. Perform. Electron. Packag. Syst. (EPEPS)*, Oct. 2018, pp. 311–313.

[19] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2019, pp. 8024–8035.

[20] R. H. Bartels and G. H. Golub, "The simplex method of linear programming using LU decomposition," *Commun. ACM*, vol. 12, no. 5, pp. 266–268, 1969.

[21] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[22] A. R. Conn, K. Scheinberg, and L. N. Vicente, *Introduction to Derivative-Free Optimization*. Philadelphia, PA, USA: SIAM, 2009.

[23] S. J. Park, B. Bae, J. Kim, and M. Swaminathan, "Application of machine learning for optimization of 3-D integrated circuits and systems," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 6, pp. 1856–1865, Jun. 2017.

[24] J. Jin, F. Feng, J. Zhang, J. Ma, and Q.-J. Zhang, "Efficient EM topology optimization incorporating advanced Matrix Padé via Lanczos and genetic algorithm for microwave design," *IEEE Trans. Microw. Theory Techn.*, vol. 69, no. 8, pp. 3645–3666, Jun. 2021.

[25] I. C. Trelea, "The particle swarm optimization algorithm: Convergence analysis and parameter selection," *Inf. Process. Lett.*, vol. 85, no. 6, pp. 317–325, Mar. 2003.

[26] W. Wang, Y. Lu, J. S. Fu, and Y. Z. Xiong, "Particle swarm optimization and finite-element based approach for microwave filter design," *IEEE Trans. Magn.*, vol. 41, no. 5, pp. 1800–1803, May 2005.

[27] S. K. Goudos and J. N. Sahalos, "Microwave absorber optimal design using multi-objective particle swarm optimization," *Microw. Opt. Technol. Lett.*, vol. 48, no. 8, pp. 1553–1558, Aug. 2006.

[28] S. Ülker, "Particle swarm optimization application to microwave circuits," *Microw. Opt. Technol. Lett.*, vol. 50, no. 5, pp. 1333–1336, 2008.

[29] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the human out of the Loop: A review of Bayesian optimization," *Proc. IEEE*, vol. 104, no. 1, pp. 148–175, Jan. 2015.

[30] N. Srinivas, A. Krause, S. M. Kakade, and M. Seeger, "Gaussian process optimization in the bandit setting: No regret and experimental design," 2009, *arXiv:0912.3995*.

[31] M. Balandat *et al.*, "BoTorch: A framework for efficient Monte-Carlo Bayesian optimization," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 21524–21538.

[32] H. M. Torun, M. Swaminathan, A. K. Davis, and M. L. F. Bellaredj, "A global Bayesian optimization algorithm and its application to integrated system design," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 26, no. 4, pp. 792–802, Apr. 2018.

[33] H. M. Torun, C. Pardue, M. L. Belleradj, A. K. Davis, and M. Swaminathan, "Machine learning driven advanced packaging and miniaturization of IoT for wireless power transfer solutions," in *Proc. IEEE 68th Electron. Compon. Technol. Conf. (ECTC)*, May 2018, pp. 2374–2381.

[34] K.-Q. Huang and M. Swaminathan, "Antennas in glass interposer for sub-THz applications," in *Proc. IEEE 71st Electron. Compon. Technol. Conf. (ECTC)*, Jun. 2021, pp. 1150–1155.

[35] H. M. Torun and M. Swaminathan, "High-dimensional global optimization method for high-frequency electronic design," *IEEE Trans. Microw. Theory Techn.*, vol. 67, no. 6, pp. 2128–2142, Jun. 2019.

[36] H. M. Torun, C. Pardue, M. L. F. Belleradj, A. K. Davis, and M. Swaminathan, "Machine learning driven advanced packaging and miniaturization of IoT for wireless power transfer solutions," in *Proc. IEEE 68th Electron. Compon. Technol. Conf. (ECTC)*, May 2018, pp. 2374–2381.

[37] J. W. Bandler, R. M. Biernacki, S. H. Chen, P. A. Grobelny, and R. H. Hemmers, "Space mapping technique for electromagnetic optimization," *IEEE Trans. Microw. Theory Techn.*, vol. 42, no. 12, pp. 2536–2544, Dec. 1994.

[38] J. E. Rayas-Sanchez, F. Lara-Rojo, and E. Martinez-Guerrero, "A linear inverse space-mapping (LISM) algorithm to design linear and nonlinear RF and microwave circuits," *IEEE Trans. Microw. Theory Techn.*, vol. 53, no. 3, pp. 960–968, Mar. 2005.

[39] M. Simsek and N. S. Sengor, *Solving Inverse Problems by Space Mapping With Inverse Difference Method*. Jan. 2010, pp. 453–460.

[40] A. Pietrenko-Dabrowska, S. Koziel, and J. W. Bandler, "Rapid microwave optimization using a design database and inverse/forward metamodels," in *IEEE MTT-S Int. Microw. Symp. Dig.*, Aug. 2020, pp. 868–871.

[41] R. Trinchero, M. A. Dolatsara, K. Roy, M. Swaminathan, and F. G. Canavero, "Design of high-speed links via a machine learning surrogate model for the inverse problem," in *Proc. Electr. Design Adv. Packag. Syst. (EDAPS)*, 2019, pp. 1–3.

[42] H. Kabir, Y. Wang, M. Yu, and Q.-J. Zhang, "Neural network inverse modeling and applications to microwave filter design," *IEEE Trans. Microw. Theory Techn.*, vol. 56, no. 4, pp. 867–879, Apr. 2008.

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

[44] K. Roy, M. A. Dolatsara, H. M. Torun, R. Trinchero, and M. Swaminathan, "Inverse design of transmission lines with deep learning," in *Proc. IEEE 28th Conf. Electr. Perform. Electron. Packag. Syst. (EPEPS)*, Oct. 2019, pp. 1–3.

[45] C. Zhang, J. Jin, W. Na, Q. J. Zhang, and M. Yu, "Multivalued neural network inverse modeling and applications to microwave filters," *IEEE Trans. Microw. Theory Techn.*, vol. 66, no. 8, pp. 3781–3797, Aug. 2018.

[46] E. Alpaydin, *Introduction to Machine Learning*. Cambridge MA, USA: MIT Press, 2014.

[47] O. W. Bhatti, N. Ambasana, and M. Swaminathan, "Inverse design of power delivery networks using invertible neural networks," in *Proc. IEEE 30th Conf. Electr. Perform. Electron. Packag. Syst. (EPEPS)*, Oct. 2021, pp. 1–3.

[48] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using real NVP," 2017, *arXiv:1605.08803*.

[49] L. Ardizzone *et al.*, "Analyzing inverse problems with invertible neural networks," 2019, *arXiv:1808.04730*.

[50] J. A. Gubner, *Probability and Random Processes for Electrical and Computer Engineers*. Cambridge, U.K.: Cambridge Univ. Press, 2006.

[51] H. Yu, H. M. Torun, M. U. Rehman, and M. Swaminathan, "Design of SIW filters in D-band using invertible neural nets," in *IEEE MTT-S Int. Microw. Symp. Dig.*, Aug. 2020, pp. 72–75.

[52] M. A. Dolatsara, H. Yu, J. A. Hejase, W. D. Becker, and M. Swaminathan, "Invertible neural networks for inverse design of CTLE in high-speed channels," in *Proc. IEEE Electr. Design Adv. Packag. Syst. (EDAPS)*, Dec. 2020, pp. 1–3.

[53] B. Gilbert, "A precise four-quadrant multiplier with subnanosecond response," *IEEE J. Solid-State Circuits*, vol. JSSC-3, no. 4, pp. 365–373, Dec. 1968.

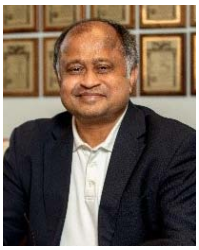
[54] J. Wholey and I. Kipnis, "Silicon bipolar active mixers," *Appl. Microw. J.*, pp. 287–293, Spring 1990.

[55] *Advanced Design System*.

[56] B. Razavi, *RF Microelectron* (Prentice Hall Communications Engineering and Emerging Technologies Series), 2nd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2011.

[57] D. M. Pozar, *Microwave Engineering*, 4th ed. Hoboken, NJ, USA: Wiley, 2012.

[58] N. Ambasana *et al.*, "Invertible neural networks for high-speed channel design & parameter distribution estimation," in *Proc. IEEE 30th Conf. Electr. Perform. Electron. Packag. Syst. (EPEPS)*, Oct. 2021, pp. 1–3.



Madhavan Swaminathan (Fellow, IEEE) is currently the John Pippin Chair in microsystems packaging and electromagnetics at the School of Electrical and Computer Engineering (ECE), a Professor in ECE with a joint appointment at the School of Materials Science and Engineering (MSE), and the Director of the 3D Systems Packaging Research Center (PRC), Georgia Tech (GT) (http://www.prc.gatech.edu), Atlanta, GA, USA. He also serves as the Site Director for the NSF Center for Advanced Electronics through Machine Learning (CAEML: https://publish.illinois.edu/advancedelectronics/) and leads the heterogeneous integration area at the SRC JUMP ASCENT Center (https://ascent.nd.edu/). Prior to joining GT, he was with IBM, New York, NY, USA, working on packaging for supercomputers. He is the author of more than 530 refereed technical publications and holds 31 patents. He is the primary author and coeditor of three books and five book chapters and the founder and the co-founder of two startup companies.

Dr. Swaminathan has served as the Distinguished Lecturer for the IEEE Electromagnetic Compatibility (EMC) Society. He is the Founder of the IEEE Conference on Electrical Design of Advanced Packaging and Systems (EDAPS), a premier conference sponsored by the IEEE Electronics Packaging Society (EPS).

Osama Waqar Bhatti (Graduate Student Member, IEEE) received the bachelor's degree in electrical engineering from the National University of Sciences and Technology, Islamabad, Pakistan, in 2017. He is currently pursuing the Ph.D. degree at the Georgia Institute of Technology, Atlanta, GA, USA.

His current research interests include designing machine learning algorithms for signal and power integrity applications.

Mr. Bhatti was a recipient of the Best Paper Award at the 22nd International Symposium on Quality Electronic Design (ISQED'21).

Yiliang Guo (Graduate Student Member, IEEE) received the B.S. degree in electrical engineering from Chongqing University, Chongqing, China, in 2018, and the M.S. degree in electrical and computer engineering from the Georgia Institute of Technology, Atlanta, GA, USA, in 2020, where he is currently pursuing the Ph.D. degree in electrical and computer engineering at the 3D Systems Packaging Research Center (PRC), advised by Prof. Madhavan Swaminathan.

His research interests include machine learning, optimization, and their applications in microwave design.

Eric Huang received the B.S. degree in mechanical engineering from the National Central University, Taoyuan, Taiwan, in 2014, and the M.S. degree in mechanical and aerospace engineering from the University of California at Los Angeles, Los Angeles, CA, USA, in 2017. He is currently pursuing the Ph.D. degree in electrical and computer engineering at the 3D Systems Packaging Research Center, Georgia Institute of Technology, Atlanta, GA, USA, advised by Prof. Madhavan Swaminathan.

His current research interests include solving optimization problems in the area of antenna, microwave, and radar. He specializes in optimization and machine learning and is interested in optimal control theory and nonlinear dynamic systems analysis.

Oluwaseyi Akinwande (Graduate Student Member, IEEE) received the bachelor's degree in electrical and electronics engineering from the University of Ibadan, Ibadan, Nigeria, in 2018, and the master's degree in electrical engineering from Auburn University, Auburn, AL, USA, in 2020. He is currently pursuing the Ph.D. degree at the Georgia Institute of Technology, Atlanta, GA, USA.

His current research interests include creating machine learning algorithms to derive models used for electronic design automation with applications in high-speed channels and microwave electronics.