Applications of Artificial Intelligence on the Modeling and Optimization for Analog and Mixed-Signal Circuits: A Review

Morteza Fayazi, *Student Member, IEEE,* Zachary Colter, Ehsan Afshari, *Member, IEEE,* Ronald Dreslinski, *Member, IEEE*

***Abstract*—Recently, there have been many studies attempting to take advantage of advancements in Artificial Intelligence (AI) in Analog and Mixed-Signal (AMS) circuit design. Automated circuit sizing optimization and improving the accuracy of performance models are the two predominant uses of AI in AMS circuit design. This paper first introduces and explains the basic concepts in AI especially the ones that are more suitable to this application. Next, it surveys some recent studies of various AI techniques for AMS circuit design. Then, it discusses the main approaches as well as the pros and cons of each method. Finally, it gives meaningful insights about the current challenges and open issues, as well as recommends approaches for specific applications.**

***Index Terms*—Analog and mixed-signal circuits, artificial intelligence, circuit optimization automation, circuit performance modeling.**

I. Introduction

CUSTOM Analog and Mixed-Signal (AMS) circuits have broad applications in wireless communication, biosensors, etc. [1]–[4]. Traditionally, the design parameters of AMS circuits e.g. transistor size and biasing are calculated manually by designers [5]. However, the complexity of physical models and severe process variations with downscaling of technology node for large AMS circuits leads to inefficiency and hardship of robust manual design especially in corner extraction [6], [7]. Moreover, generating post-layout simulation is very time-consuming. Therefore because of the growing demand of high performance, low power and time-to-market Integrated Circuits (ICs), there is a crucial need of autonomous AMS circuit design. This would catch up with the digital circuit designs, which have been automated for a long time [8], [9].

Automating AMS circuit design procedure has always been challenging as it tightly ties to human expertise and intuition to make a relationship between various parameters and performances. Given the main circuit topology, in order to satisfy the desired specifications, the circuit parameters should be chosen optimally [10]. So, this circuit optimization problem, i.e. determining the circuit parameter values to meet the required specifications, can be solved using mathematical optimization methods. As a result, a Computer-Aided Design (CAD) tool would be able to automate this circuit sizing optimization procedure leveraging techniques such as gradient-based, convex optimization, and evolutionary algorithms [11], [12]

All authors are with the Department of Electrical and Computer Science, University of Michigan, Ann Arbor, MI, 48109 USA (e-mail: fayazi@umich.edu, zcolter@umich.edu, afshari@umich.edu, rdreslin@umich.edu).

The main approaches for AMS circuit parameter search automation are deterministic techniques (e.g. Linear Programming (LP), convex Nonlinear Programming (NLP), and nonconvex Mixed-Integer Nonlinear Programming (MINLP) [13]), particle swarm intelligence, simulated annealing, evolutionary algorithm, and Bayesian Optimization (BO) [14]–[17]. Although each of these methods has its own advantages, they have some drawbacks as well. For instance, PSO suffers from a low convergence rate and BO is subjected to having a very long runtime. Furthermore, annealing can readily fall into the local minimum. Also, evolutionary algorithms are stochastic and lack reproducibility. Artificial Neural Networks (ANNs) are another promising method for AMS circuit design automation which can address the aforementioned difficulties [18].

Additionally, artificial intelligence algorithms along with other methods are suitable for yield estimation and making high-order models which can be used instead of complex physical models with a less computational cost for AMS circuit design [19]–[21]. Also, Deep Reinforcement Learning (DRL) can solve many human decision making problems in general [22]. Furthermore, AI can be leveraged to replace lengthy and costly measurements by a set of simpler measurements to ease the AMS production test [23]. The main AI challenge to have an accurate model is providing robust and enough simulation data for training sets due to essential high-dimensional variation space to model the process variation and costly simulation in a growing AMS system size. Also, providing Intellectual Properties (IPs) from different IC companies [18], [24], [25] is another barrier of dataset collection. Automatically optimized AMS circuit sizing is another application of AI. In such approaches, Electronic Design Automation (EDA) tools try combinations of design variables to find a sizing that meets the desired specifications instead of the traditional circuit sizing approach i.e. going from the target specification to the corresponding device sizes [26]. After automation of each AMS block sizing, automation of whole System on Chip (SoC) is doable with integrating these pieces [27], [28].

The layout is another important part of AMS design and there have been many recent studies on applications of AI in AMS layout design that have been reviewed thoroughly in [29], [30]. However, this paper focuses on recent applications

of AI methods in modeling and optimization of AMS circuit design. After introducing the main concepts and techniques of AI in Section II, the state-of-the-art techniques for using AI in automated AMS circuit sizing optimization and performance modeling are explained, respectively in Sections III and IV. Next, Section V discusses the main challenges and gives some ideas about future works. Finally, the paper is concluded in Section VI.

II. Artificial Intelligence & Machine Learning Main Concepts and Techniques

A. General Terminology

Artificial intelligence and Machine Learning (ML) are terms used to describe algorithms that can learn patterns without direct human involvement. Generally, these algorithms are trained, meaning that they learn patterns from some dataset, then they are used for inference. There are several common ways to train a machine learning model e.g. supervised, unsupervised, semi-supervised, and Reinforcement Learning (RL).

1) Supervised learning: Supervised learning is when the correct corresponding outputs, called labels, are known for every input [31]. This is helpful to train a model to recognize specific useful patterns. Supervised learning allows models to achieve high accuracy in many tasks, such as classification or regression, when enough data is present [32]. The largest drawback of supervised learning is that it either requires the training dataset to already have properly labeled data or it requires a significant amount of work to create and maintain a properly labeled dataset which is both time-consuming and is hard to achieve. Additionally, the designers need to be careful when choosing their training data. If this subset is not representative of the entire set, the model may output inconstant results during inference.

2) Unsupervised learning: In contrast to supervised learning, unsupervised learning does not require labeled data. This means that less effort is required to obtain proper data, but in exchange, it can be harder to train the model to create the desired output. A powerful approach with unsupervised learning is to cluster the data [33]. Clustering can lead to finding similarities between the features of various inputs to help classification or data extraction.

3) Semi-supervised learning: Semi-supervised learning, as the name implies, is a hybrid between the previous two types of learning [33]. Generally, this approach first uses unsupervised learning on a large set of unlabeled data to learn robust patterns. Then, using the labeled data, the model is trained to use those learned patterns to output relevant data. This approach, if implemented correctly, can achieve the best of both worlds. It does not need a vast set of labeled data, yet the training can still be directed. However, additional effort is needed to make sure that both parts of the network properly converge.

4) Reinforcement learning: Reinforcement learning is a version of machine learning where a software agent, a program with the ability to learn, is rewarded for certain actions [34]. Even though RL models are trained for a specific set of specifications and their reuse is not guaranteed, a major benefit of them over supervised learning is that optimal solutions to problems do not need to be known beforehand. For instance, Wang *et al.* [18] use RL to optimize the circuit parameters. They "reward" the model when it outputs a circuit with the required specifications and low power consumption and area.

5) AI model verification and performance evaluation: During training a model, the designer has to be wary of underfitting and overfitting. Underfitting happens when the model has not recognized all of the general patterns that exist in the training data. To fix this issue, the model can be trained longer on the data in order to reach a local minimum or even the global minimum. However, if the model is too simple to learn all of the patterns, then further training will not be beneficial. In this case, a larger or more complex model is needed in order to get better results. On the other hand, overfitting occurs when the model learns to identify non-generalizable patterns within the training data. For instance, a model that detects human faces would be overfitting if it could only detect faces indoors. This might happen because all of the training data with faces came from pictures taken indoors. Adding more diverse training data can help reduce this problem. Overfitting occurs more readily in larger, more complex models since they can identify and use more features compared to simpler models. To combat this, several tricks have been proposed to reduce this issue in various types of models, but there is no perfect solution to stop overfitting [35].

Once a model has been created and trained, it is necessary to verify that it predicts the output with sufficient accuracy. To verify the model, data not used in the training process is used. Testing the model on the training data only shows how well the model learns that particular data and it would be unknown if the model could accurately predict a new input. When a model is continuously modified, in order to achieve a high accuracy or a lower computation requirement, a designer may withhold another set of data for further testing. This can be done to guarantee that an increase or decrease in performance is not due to the model's hyper-parameters overfitting.

Precision, recall, accuracy, and F1-Score are measures used to evaluate the performance of a classification model. For instance, if a classification model tries to recognize "good" and "bad" candidate circuit designs [22], true positives (*TP*) denote correctly identified "good" designs. While, false positives (*FP*) result from an incorrect classification of a circuit as a "good" design. Similarly, true negatives (*TN*) refer to correctly diagnosed "bad" designs and false negatives (*FN*) represent "good" circuits incorrectly recognized as "bad" designs. Precision, recall, accuracy, and F1-Score can be calculated by $\frac{TP}{TP+FP}$, $\frac{TP}{TP+FN}$, $\frac{TP+TN}{TP+FP+TN+FN}$, and $\frac{2*\text{Precision}*\text{Recall}}{\text{Precision}+\text{Recall}}$ respectively.

K-fold cross-validation is an even more rigorous process to verify a model, when compared to the traditional testing and training split [36]. In this process, the input is split into K different subsets. The model is run K times, with every time using a different subset for the verification. In every run, all the data except for that one slice is used to train the model.

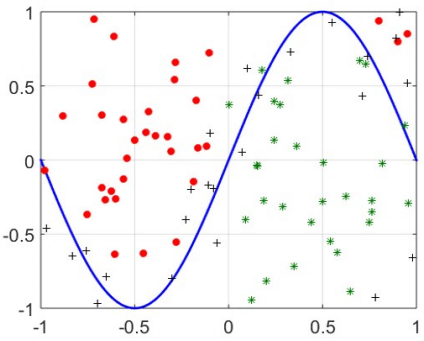


Fig. 1. An image showing a Support Vector Machine (SVM) separate two different classes (red dots and green starts) with two different features [38]. The support vectors are represented by the "+" symbol in this diagram.

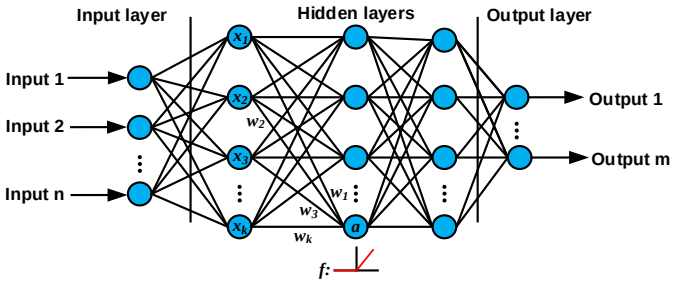


Fig. 2. An image showing an Artificial Neural Network (ANN) with several intermediate, hidden, layers [40]. If x and w are the activations and the learned weight of the previous layer that are connected to a node, the output (activation) of that node would be $a = f(\sum_{j=1}^{k} w_j x_j + b)$. In this equation, b is a learned constant bias and f is a non-linear function e.g. ReLU, sigmoid, etc.

B. Common AI & ML Models

1) Support vector machines: Many different types of artificial intelligence and machine learning models have been tested over the years. One such model, Support Vector Machine (SVM) is a supervised learning model that is mostly used for classification or regression problems. For classification, SVMs operate by finding an optimal hyperplane to separate the features of two or more classes [37]. An SVM can be seen separating two classes in Fig. 1. SVMs are used for regression by including a distance measure in the loss function. SVMs are widely praised at the start of the twenty-first century for their resilience to overfitting when compared to other ML techniques such as neural networks [37]. Today SVM models can still achieve state-of-the-art performance in certain circuit design applications such as a seizure detection sensor [20].

2) Artificial neural networks: Another type of ML model, artificial neural networks, are designed to imitate how real neurons transmit and interpret data [39]. Groups of neurons, or nodes, that are not connected with one another are called layers. Multiple layers can be used to allow complex non-linear patterns to be learned as shown in Fig. 2. The output or activation, of a single neuron in an ANN can be represented with the following equation.

$$a = f(\sum_{j=1}^{k} w_j x_j + b),$$

where k is the number of activations from the previous layer that are connected with this node. Each of these activations, x, is multiplied by a unique learned weight w, then all of these products are summed together. Next, a learned constant bias b is added to the sum before the result is sent into a non-linear function f. Many non-linear functions can be used including the Rectified Linear Unit (ReLU), the sigmoid function, and the hyperbolic tangent. ANNs have broad applications in NLP, computer architecture and circuit design [41], [42].

3) Deep neural networks: With the advent of more computational power and efficient training methods, neural networks began to get larger and deeper. A new term was coined for networks with tens to hundreds of layers, Deep Neural Networks (DNN). The layers in between the input and output layers are called hidden layers because their values are hidden to the outside world. This leads to DNNs being treated as black boxes where inputs are converted into corresponding outputs [43].

The standard implementation of a DNN has dense connections between the layers, meaning that every node in every layer connects to every node in the next layer. However, other implementations exist with various benefits. One problem with traditional dense DNNs is how quickly the memory requirement increases with the model size. Because of this, the maximum size of a traditional DNN is still limited [43]. A simple method to reduce the memory requirement would be to connect nodes in layer K to only a subset of the nodes in layer $K+1$.

4) Convolutional neural networks: Another method is to instead have a moving window of weights. In this approach, a small group of weights moves across the entire image, often with a stride of 1. Several of these groups, which are called filters, are used in order to identify many different types of features. This approach is known as a Convolutional Neural Network (CNN) [44].

CNNs have multiple layers to break down complex patterns into more manageable ones. A pooling layer is often used in conjunction with a CNN. Pooling layers condense a model-defined amount of nearby data into a representation that requires less space. A common type of pooling is max-pooling. In this type, only the maximum value within each subset of data is propagated to later layers. Pooling layers help reduce computation complexity as well as help reduce overfitting [44].

Despite the wide applications of CNNs, there are many problems that deal with data that are in a non-Euclidean structure such as chemical molecules, social networks, and functional networks of the brain. However, CNNs inputs are required to be tensor e.g. images that are modeled as 2-D structure [45]. On the other hand, the non-Euclidean data structures can be represented as graphs. In a Graph Convolutional Neural Networks (GCNN), a node is represented by aggregating its own features and the features of nodes that are connected to it [46]. Settaluri *et al.* and Kunal *et al.* [47], [48] use GCNN to identify sub-circuit structures from the schematic netlist.

Generally, CNNs work best on patterns that can be found in isolated parts of the input. For this reason, CNNs are often used for image processing [44]. But because of how efficiently CNNs use memory, they have been exploited for many other types of tasks, such as text classification and digital signal processing, which traditionally were run on Recurrent Neural Networks (RNN).

5) Recurrent neural networks: RNNs are a subset of neural networks where some data altered by the network is fed back into it. Traditionally, these networks have worked best with data that has a clear sequential order. For instance, RNNs are very effective when the inputs are time dependant, like speech recognition [49]. By using feedback, RNNs can recognize patterns over long distances or periods of time.

RNNs are generally more computationally expensive than feedforward networks. This leads to RNNs typically having less individual nodes than traditional networks. Additionally, RNNs do not work well with traditional dropout layers, layers that ignore certain nodes during training to force the usage of multiple features. For these reasons, they have a high probability of overfitting. Fig. 3 summarizes different types of neural networks. Fig. 3 d) shows an RNN. In this diagram, the inputs, X_t $\forall t \in \{1, \ldots, T\}$, are separated by time instead of space like the other models. W_{hh}, b_h is the computed value that is propagated back into the network when there is a new input X_t. For every input there is a corresponding output, O_t $\forall t \in \{1, \ldots, T\}$, that is computed from the current input and the propagated data [50].

C. Other AI Models

With the popularity of artificial intelligence many unique and specialized approaches have been proposed. Some of these approaches simply learn in different ways from more common approaches, while some of the other approaches are made for specific types of datasets.

If a dataset is sparse, meaning that most of the features are near zero, a Sparse Regression (SR) method can be used to achieve greater efficiency [52]. A sparse regression method can exploit the sparse dataset by optimizing the model to be more efficient with the use of sparse polynomials.

Another type of artificial intelligence is a family of approaches called population-based algorithms. These algorithms take a group of solutions made by candidates, agents that search for solutions, and stores them. In these algorithms, multiple agents interact and trace out multiple paths to get a population of solutions [53]. Many algorithms are population-based such as Particle Swarm Optimization (PSO) and Evolutionary Algorithms (EA).

Evolutionary algorithms take inspiration from the natural process of evolution. Like most population-based algorithms, EAs have a population where every individual represents a search point in the space of potential solutions. They also keep track of the currently known rules about their environment [12]. The members of the population undergo three different types of modifications: cross-over, mutation, and selection. These steps are repeated until a sufficient solution is reached. Cross-over is when members of the population combine with others and mutation is when members have random changes. While selection is the culling of a percentage of the population. Similar to natural evolution, the selection process is based on the members' quality score or fitness [54]. EAs can cover a vast search space when enough members of the population are active.

III. Analog and Mixed-Signal Circuit optimization

A. Overview and Main Concepts

AMS circuit designers first, decide the circuit topology [55]–[57] then optimize the corresponding design parameters e.g. component sizing, and finally generate the layout [58]. A significant amount of effort has been put into optimizing components sizing because of the large effect that it has on a circuit's performance and power usage.

1) Problem formulation: The AMS design circuit sizing optimization problem can be formulated as follows.

$$\begin{aligned} &\text{minimize } f_1(x), \ldots, f_m(x) \\ &\text{subject to: } c_i(x) < 0, \forall i \in \{1, \ldots, N_c\}, \end{aligned} \tag{1}$$

where $x \in R^d$ denotes d design variables e.g. width and length of MOS transistors, and $f_l(x)$ $\forall l \in \{1, \ldots, m\}$ are the Figure of Merit (FOM) of the AMS circuits. Each FOM can be deterministic or noisy depending on the design specification. N_c represents the total number of constraints and $c_i(x)$ corresponds to the i-th constraint e.g. $x_j \in [p_j^-, p_j^+]$. When $m \neq 1$, usually there is no best design as objectives can be conflicting and it is unlikely to optimize all of them simultaneously. The goal then would be concluding the best trade-off between a set of solutions.

2) Classical approaches: The classical AMS circuit optimization approaches can be classified into the model-based (e.g. geometric programming, SVM, ANN, Gaussian Process (GP), etc. [59]–[70]) and simulation-based methods (e.g. Simulated Annealing (SA), PSO, EA, and gradient-based local search with Multiple Starting Points (MSP) which has a better convergence rate than the others [15], [71]–[81]). Analytical manually derived or regression models with simulated data are leveraged to build global models of the FOM in the model-based approaches, while the optimization is driven directly by the circuit simulations for simulation-based methods.

The model reusability and low computational cost, especially in the case of using Electro-Magnetic (EM) components, are the main advantages of model-based approaches. However, the accuracy of these models are not usually high as the number of design parameters are usually large and object and constraint functions are highly nonlinear [17].

Recently, to combat these drawbacks, hybrid methods that combine both models have been proposed for analog circuits [82], [83] as well as mm-wave and Radio Frequency (RF) circuits [84]–[87]. These hybrid methods run simulations during the optimization procedure to update online models gradually, instead of using pre-built offline models [88]–[90]. Initially, the model is constructed by the data gathered from random sampling and it guides the selection of the next point towards more optimized performance.

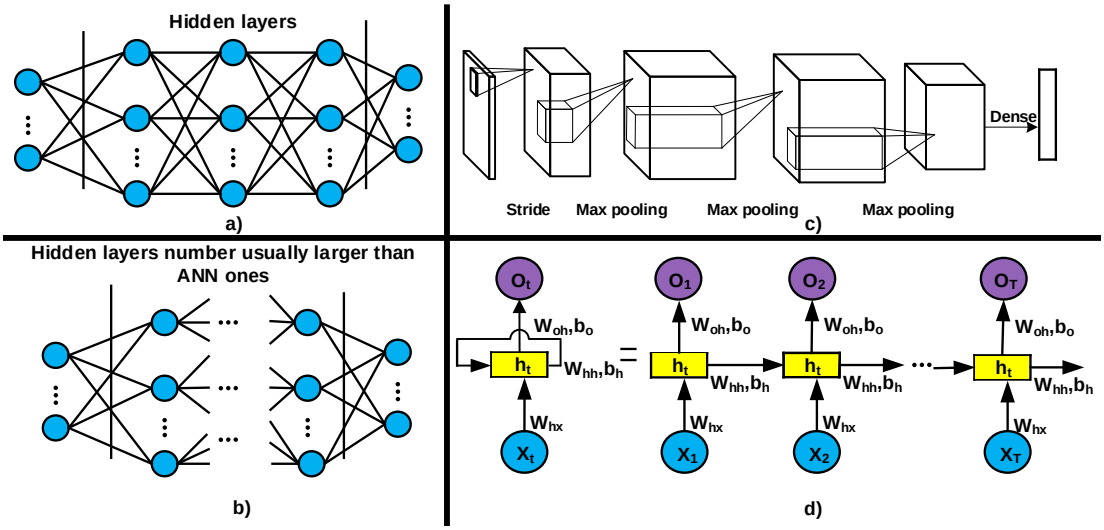


Fig. 3. Different types of neural networks: a) Artificial Neural Network (ANN) [40], b) Deep Neural Network (DNN) [40], c) Convolutional Neural Network (CNN) [51], d) Recurrent Neural Networks (RNN): W_{hh}, b_h are the computed back-propagated values. X_t are the inputs and O_t are the corresponding outputs $\forall t \in \{1, \ldots, T\}$ [50].

B. Bayesian-Based Approaches

The Bayesian optimization method after initial sampling, constructs the probabilistic surrogate model of the objective function, which is refined incrementally based on the new data that optimizes the acquisition function e.g. Expected Improvement (EI), Lower Confidence Bound (LCB), Probability of Improvement (PI) [91]–[93]. The model uncertainty is evaluated to balance the exploration, i.e. the next point tends to explore the unknown regions with high uncertainty in the surrogate model, and exploitation, i.e. the next point tends to be the optimum with high probability of prediction by the surrogate model, during the optimization [94]. The *de facto* surrogate model used in BO is the Gaussian process model [95], [96] which reduces the required number of circuit simulations and has closed-forms for both its model prediction and model uncertainty. In a GP, letting $f : X \rightarrow R$ be a black-box indicating the performance function, for any finite samples $\{x_1, \ldots, x_n\} \in \chi^n$, the vector $[f(x_1), \ldots, f(x_n)]^T$ follows joint multivariate Gaussian distribution i.e. $\begin{bmatrix} f(x_1) \\ \vdots \\ f(x_n) \end{bmatrix} \sim N(\mu, K)$, where μ is an $n \times 1$ mean vector and K is an $n \times n$ covariance matrix. The GP can be fully characterized by its mean function, $m(x)$, and its covariance function $k(x_1, x_2)$ i.e. $\mu_i = m(x_i), K_{ij} = k(x_i, x_j)$ for all $i, j \in \{1, \ldots, n\}$. K_{ij} denotes the $i-$th row $j-$th column covariance matrix element.

Let training set $D = \{X, y\}$, where $X = \{x_1, \ldots, x_N\}$, $y = \{f(x_1), \ldots, f(x_N)\}$, $m = [m(x_1), \ldots, m(x_N)]^T$, given a new point, x^*,

$$\begin{cases} k(x^*, X) = [k(x^*, x_1), \ldots, k(x^*, x_N)], \\ k(X, x^*) = k(x^*, X)^T. \end{cases}$$

Moreover, the function value, $y^* = f(x^*)$, and y follow the joint Gaussian distribution. The mean $\mu_{y^*|y}$ and the variance $\sigma^2_{y^*|y}$ can be viewed as the prediction, and the confidence of the prediction respectively.

$$\begin{cases} \mu_{y^*|y} = m(x^*) + k(x^*, X)K_N^{-1}(y - m), \\ \sigma^2_{y^*|y} = k(x^*, x^*) - k(x^*, X)K_N^{-1}k(X, x^*). \end{cases} \tag{2}$$

A Gaussian noise $\epsilon \sim N(0, \sigma_n^2)$ should be added to $f(x)$ in order to avoid overfitting of the simulation model where σ_n^2 is the variance of the Gaussian noise. If we take noise into consideration, Equation (2) can be rewritten as:

$$\begin{cases} \mu_{y^*|y} = m(x^*) + k(x^*, X)(K_N + \sigma_n^2 I)^{-1}(y - m), \\ \sigma^2_{y^*|y} = k(x^*, x^*) - k(x^*, X)(K_N + \sigma_n^2 I)^{-1}k(X, x^*). \end{cases}$$

Acquisition functions optimally balance the exploration and exploitation. The improvement of y can be formulated as

$$I(y, \tau) = \begin{cases} \tau - y & y < \tau \\ 0 & \text{otherwise}, \end{cases} \tag{3}$$

where τ is the minimal value of found $f(x)$. In the GP model, the expectation of improvement can be written as:

$$\begin{aligned} EI(x) &= \mathbb{E}[I(y, \tau)] \\ &= \int_{-\infty}^{+\infty} p(y|D, \theta)dy \\ &= (\tau - \mu(x))\Phi(\tfrac{\tau - \mu(x)}{\sigma(x)}) + \sigma(x)\phi(\tfrac{\tau - \mu(x)}{\sigma(x)}), \end{aligned}$$

where D denotes the given training set, θ is the vector of hyper-parameters, $\Phi(.)$ is the Cumulative Distribution Function (CDF) of the standard normal distribution, and $\phi(.)$ is the Probability Distribution Function (PDF) of the standard normal distribution.

Lyu *et al.* [17] propose Weighted EI (WEI) based [97], [98] BO approach for automated multi-objective AMS circuit sizing which can deal with constraints as well. It leverages a GP as its online surrogate model in contrast with the previous AMS circuit optimization methods which consider GP as either offline models or as assistance for EA. This means the surrogate model is refined incrementally when new data are observed. The reason that EI is used as the acquisition function

is that it can deal with constraints as well as the fact that it has a closed-form.

$$\omega EI(x) = \mathbb{E}[I_c(x)] = EI(x) \prod_{i=1}^{N_c} PF_i(x),$$

where $PF_i(x)$ shows the probability that each constraint, $c_i(x)$, be satisfied e.g. $c_i(x) < 0$. Moreover, they propose an algorithm for handling multi objective optimization problems.

First, using weighted min-max formulation [99], Lyu *et al.* [17] translate Equation (1) to Equation (4).

$$\begin{aligned} &\text{minimize} \max_{j \in \{1,\dots,m\}} \omega_j (f_j(x) - f_j^*) \\ &\text{subject to: } c_i(x) < 0, \forall i \in \{1, \dots, N_c\}, \end{aligned} \tag{4}$$

where f_j^* denotes the minimum of the j-th objective function. For this purpose, Equation (4) is transformed into multi single-objective optimization problems and then, the Bayesian approach can be applied for solving each. However, max-min objective function that is described in Equation (4) is not appropriate for the GP model and causes requiring more training set. For solving this issue, they transform Equation (4) to

$$\begin{aligned} &\text{minimize } \lambda \\ &\text{subject to: } \omega_j (f_j(x) - f_j^*) < \lambda, \forall j \in \{1, \dots, m\}, \\ &\qquad c_i(x) < 0, \forall i \in \{1, \dots, N_c\}, \end{aligned}$$

by leveraging a new variable λ. For evaluating their approach, they optimize a variety of circuits and EM components such as 60GHz inductor, a power amplifier, a charge pump, etc. As an example, they solve the following optimization problem for the power amplifier.

$$\begin{aligned} &\text{minimize output efficiency} \\ &\text{subject to: output power} > 23 \text{ dBm}, \\ &\qquad \text{total harmonic distortion} < 12.65 \text{ dB}. \end{aligned}$$

Although the standard BO has shown to be promising for the automated AMS circuit design, because of its sequentiality, it chooses only one point at each iteration by optimizing the acquisition function. Lyu *et al.* [5] leverage the idea of selecting a batch of points at each iteration [100]–[102] and propose a parallelized BO algorithm via Multi-objective Acquisition Ensemble (MACE) to accelerate the optimization. MACE chooses the best trade-off between acquisition functions by selecting multiple of them and capturing the Pareto front of these functions after the GP model is updated at each iteration. They test their approach by applying it to an operational amplifier, a class-E power amplifier, etc. For instance, they define a $FOM = -3 \times$*power added efficiency*$-$*output power* and optimize it.

Yield determines the robustness of a design under process and condition variation [103]. SRAMs need extremely high yield, e.g. around 10^{-7} failure rate, which makes the conventional yield estimation techniques such as Monte Carlo-based and corner-based approaches [104], [105] computationally expensive. Yield is inherently a stochastic function which makes it impossible to be accessed directly, estimated by yield analysis tools, or optimized by many efficient optimization algorithms such as gradient-based techniques. All conventional yield optimization approaches reduce the computational costs instead of decreasing the number of yield estimations. Wang *et al.* [106] apply BO optimization using the GP model with an EI acquisition function to reduce the number of yield estimations invoking during the optimization procedure. They test their approach on a few analog circuits such as a low noise amplifier, and a comparator as well as an SRAM, etc. It is worth mentioning that an accurate SRAM yield estimator should consider the impacts of peripherals such as the sense amplifier into account as the SRAM performance is related to them in addition to the local process and mismatch variations.

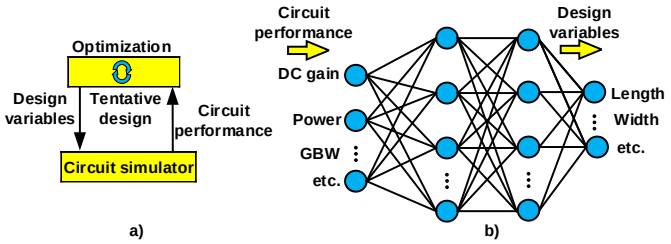


Fig. 4. The different methods of AMS circuit sizing: a) Conventional optimization-based sizing (inverse approach) i.e. from design variables to circuit performances in an iterative loop; b) Artificial Neural Networks (ANNs) (direct approach) i.e. from circuit performances to design variables [26].

C. Multiple Starting Point Approach

MSP optimization generally begins by randomly selecting large amounts of starting points that cover the design space. This first step is called the global phase. Afterward, in the local phase, an efficient local search such as Sequential Quadratic Programming (SQP) [107] is applied to each starting point found in the global phase.

MSP can successfully approximate the global optimum by choosing the best local optimum. Yang *et al.* [108] improve the inherent large MSP computation time by proposing smart-MSP. A heuristic-biased starting point selection is implemented in the global phase to find the starting points that are likely to be close to optimums instead of using randomly selected points. Furthermore, an intermediate phase is added where sparse regression is applied to predict the circuit performances around the starting points. Moreover, in the local phase, model optimums are used as the starting points for SQP and coupled with Probabilistic-TABU (P-TABU) approach [109], [110] to improve the efficiency of local searches. For evaluating their approach, they optimize a variety of circuits such as an amplifier, a charge pump, a Voltage Control Oscillator (VCO), etc. in different technology nodes.

D. Neural Network Based Approaches

In the conventional circuit sizing approaches, designers or EDA tools try to find the circuit parameters in an iterative loop while using a simulator in each iteration to evaluate the design. In order to avoid time-consuming simulations, several studies

[111]–[113] have used ANN models to replace and complement the SPICE simulator with an approximated model. They implement the simulator in later stages just to maintain the accuracy. Even though using an ANN instead of a simulator saves time considerably, the training set of such an ANN model should cover the entire design space which consumes a lot of resources. On the other hand, Lourenço *et al.* [26], as it is illustrated in Fig. 4, train an ANN model to directly size and optimize the circuit for given specifications instead of invoking the simulator many times in an iterative loop to achieve the optimized sizing. Although they use circuit sizing solutions from previous optimizations as the training set, they are able to optimize circuit sizes for specifications outside the ones in the training dataset. To have a better NN model, Lourenço *et al.* [26] take into account the following considerations for selecting the ANN hyperparameters e.g. number of layers, number of nodes in each layer, etc. First, in order to have a rich encoding, the number of nodes in each layer is increased in the first layers then is decreased toward the output layer. Second, to find a less complicated model, it is better to design it for having a low error model during the training and compensate the overfitting using L2 regularization [114]. Third, for achieving a model which gives the best cross-validation score, it is necessary to explore the hyperparameter space. Two general approaches are grid search which considers all combinations of the specified hyperparameters and random search which takes samples from a hyperparameter space with a specified distribution [115]. In their ANN approach, the circuit specifications are given to input nodes and output nodes determine the circuit sizing.

Although it successfully predicts the analog IC sizing for the target performance, its cost for providing training data is still too high, which makes it impractical. Pan *et al.* [116] propose a performance exploration method to tackle the major evolutionary-based approaches shortcoming, low accuracy, while significantly speeding up the runtime. For this purpose, they apply Bayesian regression to better model the device variables and use an SVM to increase the model performance space. In other words, they replace the performance evaluation process with SVM predictions. However, they leverage supervised learning which requires a large dataset. This dataset is difficult to obtain because most analog IPs are not available to the public. In order to evaluate their approach, they compare the specifications of an operational amplifier and a radio-frequency distributed amplifier e.g. voltage gain, DC power, bandwidth, etc. optimized by their method with [117], [118].

Wang *et al.* [18] leverage RL which learns to efficiently optimize the transistor parameters automatically without any prior knowledge about circuit design rules. In each iteration, after observations, including monitoring DC operating points, AC magnitude, and phase responses, they change the circuit parameters based on the simulator results. Then, a reward would be received to optimize the desired FOM. They compare their approach for optimizing an amplifier and meet the specifications constraints such as the specified bandwidth, gain, power, area, etc. with other methods such as a human expert, [5], etc. The results show that the other compared methods either are not able to meet the constraints at the same runtime or they have a less efficient design in comparison to [18]. Hakhamaneshi *et al.* [22] find the optimal size of analog circuits by predicting the feasibility of a design using a DNN classifier. They propose a framework that in each iteration utilizes an evolutionary algorithm to create new candidate designs and leverage a DNN classifier to recognize "bad" new offspring by comparing with a reference design that is chosen from the previous "good" population. By passing just these high quality samples to a layout-aware design methodology such as BAG [119], it achieves more than a $200\times$runtime improvement in comparison to just an evaluation method without the DNN classifier.

E. Other Approaches & Comparison

The computational complexity of the GP with pre-defined stationary kernel is $O(N^3)$ and $O(N^2)$ for training and prediction respectively where N is the number of training data. Zhang *et al.* [8] propose a neural-network-based BO approach which extracts "good" features and then define the GP using the extracted features [120], [121]. As neural networks are appropriate for providing high-quality feature extraction and the GP kernel function is similar to an inner product of nonlinear feature maps, they automatically learn a kernel function of the GP and reduce the computational complexity to $O(N)$ and a constant for training and prediction respectively.

Zhang *et al.* [20] relax the performance required, e.g. noise, INL, power of the frontend Analog-to-Digital Converter (ADC) and analog multiplier, in a sensor for seizure detection using an SVM classifier. The error-aware model is a model that its training data comes from a non-ideal system. Despite the presence of significant errors because of nonideality, error-aware models can achieve high accuracy in classifications [122]. In this regard, instead of having high power, high accuracy ADC for data conversion, the data conversion's errors are compensated by a simple non-ideal ADC followed by an appropriately trained error-aware SVM model classifier. It is worth mentioning that the SVM model is preloaded offline to the chip from the trained data.

Table I summarizes the different methods of AI-based AMS circuit sizing optimization approaches.

IV. Analog and Mixed-Signal Circuit Performance Modeling

A. Overview and Main Concepts

The main goal of performance modeling is finding f as a function of device-level variations, $\mathbf{x}$, and device-level operation point to approximate the Performance of Interest (PoI). The PoI is represented with the variable y. An example of device-level variations is $\Delta_{V_{TH}}$. DC bias voltage and voltage gain of an operational amplifier are examples of device-level operations and a performance model respectively. In other words, performance modeling tries to find f which establishes a mapping from $\mathbf{x}$ to y such that:

$$y \approx f(\mathbf{x}) = \alpha.\mathbf{g}(\mathbf{x}) \tag{5}$$

Equation (5) shows an inner product of $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_M]$, the model coefficients vector,

TABLE I
EXAMPLES OF AMS CIRCUIT SIZING OPTIMIZATION METHODS.†

Reference	AI Method	Surrogate model & acquisition function	Applications & advantages	Test results		
				# design variables	Compared algorithms	Advantage
[17]	BO	Online GP & WEI	Handling multi-objective optimizations and optimization constraints	$3-36$	[87], [75], [72], [71], [15]	Higher accuracy, fewer training number
[5]	BO	GP & multi-functions	Multi-objective acquisition function selection, parallelized BO	$10-12$	[123], [124], [100]	Higher accuracy, fastest convergence rate
[106]	BO	GP & EI	Yield optimization	$6-24$	[125], [103]	Higher accuracy, fewer training number
[8]	BO & DNN	GP & WEI	Low computation complexity, handling optimization constraints	$10-36$	[17], [72], [87]	Higher accuracy, fewer training number
[22]	EA & DNN	-	High sample efficiency	21	EA	Much faster runtime
[108]	MSP	-	Significantly fast	$11-36$	[78], [80], [90], [72], [15], [71]	Fewer training number, faster runtime
[116]	BR & SVM	-	Short runtime	-	[117], [118]	Higher accuracy, much faster runtime
[18]	RL	-	High sample efficiency	-	Human expert, [5] random search	Higher accuracy
[20]	SVM	-	Relaxing the circuit performance for an ADC	-	-	-

† Abbreviation list: BO: Bayesian Optimization, GP: Gaussian Process, WEI: Weighted Expected Improvement, EI: Expected Improvement, DNN: Deep Neural Network, EA: Evolutionary Algorithms, MSP: Multiple Starting Points, BR: Bayesian Regression, RL: Reinforcement Learning, SVM: Support Vector Machine, ADC: Analog-to-Digital Converter.

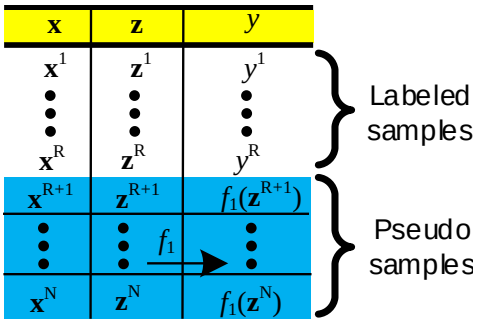


Fig. 5. The CL-BMF procedure [126]. After fitting the less complex model, f_1, using a small number of labeled samples collected by simulations i.e. $[(\mathbf{x}^1,\mathbf{z}^1,y^1),\ldots,(\mathbf{x}^R,\mathbf{z}^R,y^R)]$, a set of pseudo samples i.e. $[(\mathbf{x}^{R+1},\mathbf{z}^{R+1},f_1(\mathbf{z}^{R+1})),\ldots,(\mathbf{x}^N,\mathbf{z}^N,f_1(\mathbf{z}^N))]$ are generated with almost no cost. The more complex model, f, is trained then with all the labeled and the pseudo samples i.e. $[(\mathbf{x}^1,\mathbf{z}^1,y^1),\ldots,(\mathbf{x}^R,\mathbf{z}^R,y^R),(\mathbf{x}^{R+1},\mathbf{z}^{R+1},f_1(\mathbf{z}^{R+1})),\ldots,(\mathbf{x}^N,\mathbf{z}^N,f_1(\mathbf{z}^N))]$.

and $\mathbf{g}(\mathbf{x}) = [g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots, g_M(\mathbf{x})]^T$, the basis functions vector. Linear and quadratic polynomials are examples of these basis functions. Finding the model coefficients is the only way for solving Equation (5). One way for doing so is the traditional least-squares fitting approach which collects N sampling points of $\mathbf{x}$ and y where $\mathbf{y} = [y^1, y^2, \ldots, y^k]^T$. Then, it solves $\min_{\alpha} \|\mathbf{y} - \mathbf{G} \times \alpha^T\|_2^2$. $\mathbf{G}$ is a $N \times M$ matrix where M denotes the total number of basis functions and $\|\cdot\|_2^2$ stands for the L_2-norm.

Providing a large number of samples in today's high dimensional variation space to fit a complex AMS circuit model is time-consuming and expensive especially when it includes post-layout simulation. Several studies [127]–[133] have proposed to tackle this issue. These approaches leverage the Coefficient Side Information (CSI) which is an extra prior knowledge related to the model coefficients. An example of such knowledge is the fact that the coefficients are sparse, meaning that most of them are close to zero [128]–[131], [133]. A model that captures this inherent sparsity avoids overfitting to noisy data in addition to decreasing the computational cost.

In order to make sparse regression more tractable, two broad approaches, relaxation-based and heuristics, have been proposed. Relaxation-based approaches usually use Lasso and ridge regression [134], [135] while a Laplace and a Gaussian distributions have been applied prior on the model coefficients respectively. The main disadvantages of relaxation-based methods are penalizing model coefficients with high values, not being able to identify a subset of important variables i.e. true variable selection, and not directly capturing uncertainty. On the other hand, while heuristics approaches directly perform variable selection and clearly identify the important features, they may be unstable when the number of labeled samples is very small and some information can be lost during the feature selection process [21]. Although sparse regression has a lot of practical applications, it still requires a large training set for a typical high-dimensional AMS circuit performance model.

Afcan *et al.* [29] review the old studies of applying machine learning on analog circuit modeling which mainly leverage SVM and ANN. However, recent studies focus more on Bayesian model fusion and semi-supervised learning approaches.

B. Bayesian Model Fusion Based Approaches

There are several studies which have applied Bayesian Model Fusion (BMF) [24], [126], [136]–[141] to accurately estimate the parametric and the statistical distribution of circuit performance for both pre-silicon verification and post-silicon validation. Wang *et al.* [138] leverage the idea of BMF for

performance modeling. The fusion in BMF means passing information as the prior knowledge from a simple early-stage model, e.g. schematic-level, to a late-stage performance model, e.g. post-layout, to reduce the expensive late-stage modeling cost. Hence, BMF after approximating the early-stage performance model using simulated data, generates sample points for the late-stage model for the same performance using this early-stage model. This prior knowledge is then combined with very few late-stage sampling points to solve the late-stage model coefficients via Bayesian inference [142]. As a result, only a small number of sampling points are required to fit a high-dimensional late-stage model.

In addition to AMS performance modeling, BMF has applications on other AMS design aspects. In order to have a reliable circuit, yield should be estimated and optimized accurately by designers. The conventional yield estimation approaches e.g. Monte Carlo based and corner-based approaches require collecting many simulation samples to accurately estimate the performance which makes them extremely computationally expensive. Li *et al.* [136] propose BMF for AMS circuit parametric yield estimation. Fang *et al.* [140] use BMF to estimate Bit Error Rate (BER) in high-speed I/O links. While all the aforementioned studies consider modeling a single performance, Huang *et al.* [139] leverage BMF for approximating multiple correlated performances assuming the probability distribution of these metrics is jointly Gaussian.

Wang *et al.* [126] use the concept of co-learning [143] and propose Co-Learning BMF (CL-BMF) to even reduce the training set in the early-stage model. They pass the knowledge from a less complex model which predicts the performance of a circuit with a small trained dataset to a more complex model that predicts the performance of the same circuit. Using this prior knowledge, the complex model skips the initial steps of training which saves time. In other words, the more complex model reuses the information generated accurately from the less complex one for its training set instead of relying on the expensive training data only. Fig. 5 depicts CL-BMF steps where $\mathbf{z}$ is a vector of performance metrics that 1) is low-dimensional, inexpensive to simulate, 2) does not share any common element with $\mathbf{x}$, and 3) $y \approx f_1(\mathbf{z})$ such that f_1 is a less complex model and needs smaller training set than f in Equation (5).

C. Semi Supervised Learning Approaches

All the aforementioned approaches are classified as supervised learning as they use only labeled training sets. To reduce the amount of work required for gathering the labeled data, several studies have shifted gears toward semi-supervised learning [21], [24], [52], [141]. This is used because less labeled data is required to build accurate models. Alawieh *et al.* [24], [141] propose Bayesian co-learning hierarchical performance modeling. It is called as such because they partition the entire circuit into multiple blocks, e.g. Low Noise Amplifier (LNA), mixer, etc. for a transceiver circuit and use the concept of co-learning. In this semi-supervised learning approach, the block-level performance can be modeled with low computational cost. After partitioning the circuit, they map block-level performance metrics, e.g. gain of an LNA, to the PoI at the circuit level using a low-dimensional model. They train both block-level and mapping models on a small number of labeled samples and these models generate pseudo samples of un-labeled data at almost zero cost. Finally, the high-dimensional circuit level model is fitted by combining the aforementioned low-dimensional block-level and mapping models. This leverages both labeled and unlabeled data at a very low cost. Although this approach remarkably reduces the modeling cost, its application is limited to circuits where a block-level partitioning is possible and intuitive.

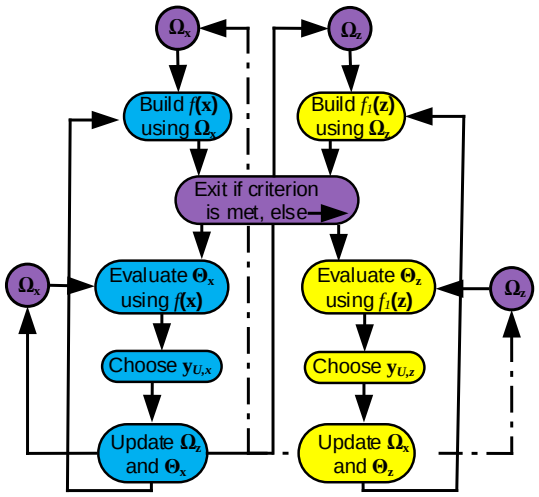


Fig. 6. The semi-supervised co-learning framework [52]. $\mathbf{\Omega}$ and $\mathbf{\Theta}$ denote training and unlabeled dataset respectively, while $\mathbf{y}_{U,x}$ and $\mathbf{y}_{U,z}$ represent pseudo labels for unlabeled samples $\mathbf{\Theta_x}$ and $\mathbf{\Theta_z}$ respectively. After building f and f_1 models from the training set, highly-confident pseudo samples are chosen from the unlabeled dataset. These highly-confident pseudo samples are used to update the training set of the other model at the next iteration.

Alawieh *et al.* [52] propose a semi-supervised co-learning-based framework without any assumption about the AMS circuit structure that leverages two views for each device in the circuit. The device level variations e.g. $\Delta_{V_{TH}}$, and independent random Process Variables (PV). They use a small number of labeled samples to build an initial model for each view. Then, at each iteration, highly-confident pseudo samples which are generated by the other model are combined with the small number of available labeled samples to update the model. In other words, they build $y \approx f_1(\mathbf{z}) = \beta.\mathbf{c}(\mathbf{z})$ model along with $y \approx f(\mathbf{x}) = \alpha.\mathbf{g}(\mathbf{x})$ where y is the PoI, $\mathbf{x}$ and $\mathbf{z}$ are vectors containing the device level variations and PV respectively. Then, to update, i.e. make a better estimation of $y \approx f(\mathbf{x}) = \alpha.\mathbf{g}(\mathbf{x})$, they use the combination of pseudo samples generated by $y \approx f_1(\mathbf{z}) = \beta.\mathbf{c}(\mathbf{z})$ and a small number of labeled samples. This procedure is shown in Fig. 6.

D. Other Approaches & Comparison

Alawieh *et al.* [21] utilize Bayesian spike and slab feature selection techniques [144] to tackle relaxation-based and heuristic methods in sparse regression. Spike and slab models explicitly classify variables as important or non-important independently of the feature selection mechanism. They build

TABLE II
EXAMPLES OF AMS CIRCUIT MODELING METHODS. †

	AI Method	Prior knowledge source/ CSI	Application	Modeling Scope
Li *et al.* [136]	BMF	Schematic	Yield estimation	Post-layout
Wang *et al.* [138]	BMF	Schematc	Performance modeling	Post-layout
Wang *et al.* [126]	CL-BMF	Simple performance model	Performance modeling	Schematic
Alawieh *et al.* [141]	Hierarchical Bayesian co-learning	Block-level performance, Sparsity	Performance modeling	Schematic
Alawieh *et al.* [52]	Semi-supervised co-learning	Sparsity	Performance modeling	Schematic
Alawieh *et al.* [21]	Bayesian spike & slab feature selection	Spike and slab sparsity	Performance modeling	Schematic

† Abbreviation list: CSI: Coefficient Side Information, BMF: Bayesian Model Fusion, CL-BMF: Co-Learning Bayesian Model Fusion.

a hierarchical Bayesian framework to learn the importance and coefficients values. Leveraging a Gibbs sampler [145] enables directly capturing the estimation uncertainty, unlike optimization-based methods.

Table II summarizes the different methods of AI-based AMS circuit modeling methods.

V. FUTURE WORKS & CHALLENGES

As mentioned earlier, one of the main challenges of AMS circuit automation is its dependency on circuit knowledge expertise. This fact leads to shifting gears from viewing an AMS circuit design problem as a physics-based question to a math problem. Viewing an AMS circuit optimization task as a mathematical optimization problem, especially an AI problem, causes significant breakthroughs in the circuit design automation. However, a lot of circuit insight, that would help make a more efficient design, has been thrown away. Leveraging this circuit knowledge helps in both circuit optimization and performance modeling by either having a more optimized algorithm or by decreasing the amount of required training data.

While using a math model, instead of a highly complex physical model, brings many advantages, some trade-offs are made. The physical model still contains some information that the pure math model has trouble identifying. Therefore having a hybrid approach could potentially lead to better results. For instance, having a very simple circuit design insight which can be used as the design rule of thumbs can guide a better search to find the most optimum design. Moreover, it can lead to narrowing down the required training set and providing a ground truth to evaluate the AI model.

As an example, consider the operational amplifier shown in Fig. 7. Some of simple circuit design principles for this circuit can be written as follows.

- In order to eliminate the common-mode gain, which contradicts the purpose of having a differential circuit we should have $M_1 = M_2$, $M_3 = M_4$, $M_5 = M_6$, and $M_7 = M_8$.
- The gain is independent of M_7 and M_8 widths. Moreover, in a well-designed circuit, M_{10} and M_{11} have a minimal contribution to the amplifier gain as they work as a unity-gain buffer.
- As far as the input-referred noise is concerned, M_1, M_2, M_7, and M_8 are the main contributors. The remaining devices have minimal to no effect.
- Unity-gain bandwidth, which denotes the maximum frequency that the circuit can amplify a voltage, to a first-order, equals $\frac{g_{m1}}{C}$. This means that it is independent from the other design parameters.
- Slew rate is a function of V_{b1}, M_9, and C. The rest of the circuit does not play a role.

These very simple principles can significantly reduce the design space in the circuit design automation. For example, if the tool knows the slew rate is just a function of three parameters, i.e. V_{b1}, M_9, and C, its design space would shrink considerably. Similarly, determining parameters that are independent of gain, input-referred noise, and unity-gain bandwidth avoids the necessity to optimize them, making the optimization model more efficient. Moreover, as it is described in Section III and IV, one technique that accelerates the searching procedure of finding the optimum sizing or performance model with less training set is the ability to guide the prediction procedure. For instance, Hakhamaneshi *et al.* [22] detects "good" and "bad" candidates to avoid processing all of them in order to guide the learning procedure. A similar guidance mechanism can be applied using these simple circuit design principles.

Training can be guided by simple circuit design principles such as methods to avoid common-mode gain or which attributes are proportional or inversely proportional to specifications. It is worth mentioning that applying these circuit design principles is not in contradiction with the goal of artificial intelligence and circuit design automation. These are just very simple rules for a better guidance which significantly increases the design procedure efficiency while the rest of the design would be still using pure artificial intelligence and circuit design automation techniques.

Moreover, even it is possible that the tool learns these circuit principles which opens a new window for the future works. As it is explained in Section IV, the idea of hierarchical learning, e.g. [24], [141] can be very useful as it breaks down a large design space into multiple smaller ones. In this case, at the first level of the hierarchy, the tool can learn the effects of each parameter or a group of parameters on every specification, then in the second level of the hierarchy, the learned principles of the first level can be leveraged to design the circuit.

The other perspective for the future works would be using AI to improve the efficiency or relaxing the complexity of a specific AMS circuit design. Most studies so far have focused on either AMS circuit optimization or performance modeling in general. However, narrowing down the scope for a specific circuit type e.g. Phase Lock Loop (PLL), ADC, etc. and

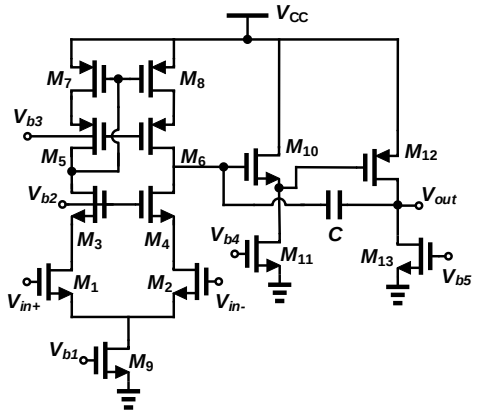


Fig. 7. An example of an operational amplifier.

then analyzing them can be more efficient. Moreover, each of these circuits could benefit from AI in a unique way. For example, in [20], applying AI to an ADC leads to a more efficient and less complex design. Another application of AI for AMS circuit design would be using a simple AI model to detect and remove noise instead of a complex conventional circuit. In order to automate this procedure, one of the main challenges is first identifying and classifying different circuit types from their netlist. This task would be an extrapolation of what Liou *et al.* [146] propose for classifying sub-circuits into digital and analog. This can be considered as one example of AMS circuits self-healing. Using AI for automating 1) diagnosis failure and degraded specifications, and 2) solving them can address many circuit design challenges [147].

VI. Conclusion

This paper studies the recent applications of artificial intelligence for AMS modeling and optimization. After reviewing the main concepts and techniques of AI, previous works have been divided into two main classes i.e. circuit optimization and performance modeling. The main motivation of each work along with their advantages and disadvantages are explained. Finally, the current challenges of using AI in AMS circuit design and future works have been discussed.

Acknowledgment

The authors would like to thank Mohammadhasan Fayazi and Javad Bagherzadeh for their helpful discussions.

References

[1] B. Razavi and R. Behzad, *RF microelectronics*. Prentice Hall New York, 2012, vol. 2.

[2] A. Turner, I. Karube, and G. S. Wilson, *Biosensors: fundamentals and applications*. Oxford university press, 1987.

[3] B. Rostami, F. Shanehsazzadeh, and M. Fardmanesh, "Fast fourier transform based ndt approach for depth detection of hidden defects using hts rf-squid," *IEEE Transactions on Applied Superconductivity*, vol. 28, no. 7, pp. 1–6, 2018.

[4] A. Kosari, M. Moosavifar, and D. D. Wentzloff, "A 152muw- 99dbm bpsk16-qam ofdm receiver for lpwan applications," in *2018 IEEE Asian Solid-State Circuits Conference (A-SSCC)*. IEEE, 2018, pp. 303–306.

[5] W. Lyu, F. Yang, C. Yan, D. Zhou, and X. Zeng, "Batch bayesian optimization via multi-objective acquisition ensemble for automated analog circuit design," in *International Conference on Machine Learning*, 2018, pp. 3306–3314.

[6] X. Li, J. Le, and L. T. Pileggi, *Statistical performance modeling and optimization*. Now Publishers Inc, 2007, vol. 3.

[7] H. Zhang, T.-H. Chen, M.-Y. Ting, and X. Li, "Efficient design-specific worst-case corner extraction for integrated circuits," in *2009 46th ACMIEEE Design Automation Conference*. IEEE, 2009.

[8] S. Zhang *et al.*, "Bayesian optimization approach for analog circuit synthesis using neural network," in *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2019.

[9] A. Amarnath, J. Bagherzadeh, J. Tan, and R. G. Dreslinski, "3dtube: A design framework for high-variation carbon nanotube-based transistor technology," in *2019 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*. IEEE, 2019, pp. 1–6.

[10] P. P. Prajapati and M. V. Shah, "Two stage cmos operational amplifier design using particle swarm optimization algorithm," in *2015 IEEE UP Section Conference on Electrical Computer and Electronics (UPCON)*. IEEE, 2015, pp. 1–5.

[11] A. R. Conn *et al.*, "Gradient-based optimization of custom circuits using a static-timing formulation," in *Proceedings of the 36th annual ACM/IEEE Design Automation Conference*, 1999, pp. 452–459.

[12] T. Back, *Evolutionary algorithms in theory and practice evolution strategies, evolutionary programming, genetic algorithms*. Oxford university press, 1996.

[13] M.-H. Lin, J.-F. Tsai, and C.-S. Yu, "A review of deterministic optimization methods in engineering and management," *Mathematical Problems in Engineering*, vol. 2012, 2012.

[14] P. K. Rout, D. P. Acharya, and G. Panda, "A multiobjective optimization based fast and robust design methodology for low power and low phase noise current starved vco," *IEEE Transactions on Semiconductor Manufacturing*, vol. 27, no. 1, pp. 43–50, 2013.

[15] R. Phelps, M. Krasnicki, R. A. Rutenbar, L. R. Carley, and J. R. Hellums, "Anaconda: simulation-based synthesis of analog circuits via stochastic pattern search," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, no. 6, 2000.

[16] B. Liu, F. V. Fern'andez, G. Gielen, R. Castro-L'opez, and E. Roca, "A memetic approach to the automatic design of high-performance analog integrated circuits," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 14, no. 3, pp. 1–24, 2009.

[17] W. Lyu *et al.*, "An efficient bayesian optimization approach for automated optimization of analog circuits," *IEEE Transactions on Circuits and Systems I Regular Papers*, vol. 65, no. 6, 2017.

[18] H. Wang, J. Yang, H.-S. Lee, and S. Han, "Learning to design circuits," *arXiv preprint arXiv1812.02734*, 2018.

[19] Y. Lin, M. B. Alawieh, W. Ye, and D. Z. Pan, "Machine learning for yield learning and optimization," in *2018 IEEE International Test Conference (ITC)*. IEEE, 2018, pp. 1–10.

[20] J. Zhang, L. Huang, Z. Wang, and N. Verma, "A seizure-detection ic employing machine learning to overcome data-conversion and analog-processing non-idealities," in *2015 IEEE Custom Integrated Circuits Conference (CICC)*. IEEE, 2015, pp. 1–4.

[21] M. B. Alawieh, S. A. Williamson, and D. Z. Pan, "Rethinking sparsity in performance modeling for analog and mixed circuits using spike and slab models," in *2019 56th ACMIEEE Design Automation Conference (DAC)*. IEEE, 2019, pp. 1–6.

[22] K. Hakhamaneshi, N. Werblun, P. Abbeel, and V. Stojanovi'c, "Analog circuit generator based on deep neural network enhanced combinatorial optimization," in *Proceedings of the 56th Annual Design Automation Conference 2019*, 2019, pp. 1–2.

[23] M. J. Barragan *et al.*, "On the use of causal feature selection in the context of machine-learning indirect test," in *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2019.

[24] M. B. Alawieh, F. Wang, and X. Li, "Efficient hierarchical performance modeling for analog and mixed-signal circuits via bayesian co-learning," *IEEE TCAD*, vol. 37, no. 12, pp. 2986–2998, 2018.

[25] R. Dreslinski *et al.*, "Fully-autonomous soc synthesis using customizable cell-based synthesizable analog circuits," University of Michigan Ann Arbor United States, Tech. Rep., 2019.

[26] N. Lourencco *et al.*, "On the exploration of promising analog ic designs via artificial neural networks," in *2018 15th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*. IEEE, 2018.

[27] T. Ajayi *et al.*, "Fully autonomous mixed signal soc design & layout generation platform," 2020.

[28] T. Ajayi *et al.*, "An open-source framework for autonomous soc design with analog block generation," in *2020 IFIP/IEEE 28th International Conference on Very Large Scale Integration (VLSI-SOC)*. IEEE, 2020.

[29] E. Afacan, N. Lourenço, R. Martins, and G. Dündar, "Machine learning techniques in analog/rf integrated circuit design, synthesis, layout, and test," *Integration*, 2020.

[30] H. Chen *et al.*, "Challenges and opportunities toward fully automated analog layout design," *Journal of Semiconductors*, vol. 41, no. 11, p. 111407, 2020.

[31] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas, "Supervised machine learning a review of classification techniques," *Emerging artificial intelligence applications in computer engineering*, vol. 160, no. 1, pp. 3–24, 2007.

[32] P. Chaovalit and L. Zhou, "Movie review mining a comparison between supervised and unsupervised classification approaches," in *Proceedings of the 38th annual Hawaii international conference on system sciences*. IEEE, 2005, pp. 112c–112c.

[33] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv1511.06434*, 2015.

[34] R. S. Sutton, A. G. Barto *et al.*, *Introduction to reinforcement learning*. MIT press Cambridge, 1998, vol. 135.

[35] H. Jabbar and R. Z. Khan, "Methods to avoid over-fitting and under-fitting in supervised machine learning (comparative study)," *Computer Science, Communication and Instrumentation Devices*, 2015.

[36] J. D. Rodriguez, A. Perez, and J. A. Lozano, "Sensitivity analysis of k-fold cross validation in prediction error estimation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 3, 2009.

[37] S. R. Gunn *et al.*, "Support vector machines for classification and regression," *ISIS technical report*, vol. 14, no. 1, pp. 5–16, 1998.

[38] S.-i. Amari and S. Wu, "Improving support vector machine classifiers by modifying kernel functions," *Neural Networks*, vol. 12, no. 6, 1999.

[39] A. K. Jain, J. Mao, and K. M. Mohiuddin, "Artificial neural networks a tutorial," *Computer*, vol. 29, no. 3, pp. 31–44, 1996.

[40] F. Bre, J. M. Gimenez, and V. D. Fachinotti, "Prediction of wind pressure coefficients on building surfaces using artificial neural networks," *Energy and Buildings*, vol. 158, pp. 1429–1441, 2018.

[41] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[42] C. Raffel *et al.*, "Exploring the limits of transfer learning with a unified text-to-text transformer," *arXiv preprint arXiv:1910.10683*, 2019.

[43] W. Liu *et al.*, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11–26, 2017.

[44] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, "Face recognition a convolutional neural-network approach," *IEEE transactions on neural networks*, vol. 8, no. 1, pp. 98–113, 1997.

[45] R. Li, S. Wang, F. Zhu, and J. Huang, "Adaptive graph convolutional neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[46] Z. Wu *et al.*, "A comprehensive survey on graph neural networks," *IEEE transactions on neural networks and learning systems*, 2020.

[47] K. Settaluri and E. Fallon, "Fully automated analog sub-circuit clustering with graph convolutional neural networks," in *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2020, pp. 1714–1715.

[48] K. Kunal *et al.*, "Gana: Graph convolutional network based automated netlist annotation for analog circuits," in *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2020, pp. 55–60.

[49] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent neural network regularization," *arXiv preprint arXiv1409.2329*, 2014.

[50] A. Mobiny, M. Najarian, and J. Jahanipour, "Vanilla rnn for digit classification," http://www.easy-tensorflow.com/tf-tutorials/recurrent-neural-networks/vanilla-rnn-for-classification, last accessed 2020-10-27.

[51] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[52] M. B. Alawieh, X. Tang, and D. Z. Pan, "S2-pm semi-supervised learning for efficient performance modeling of analog and mixed signal circuits," in *Proceedings of the 24th Asia and South Pacific Design Automation Conference*, 2019, pp. 268–273.

[53] A. E. Babalola, B. A. Ojokoh, and J. B. Odili, "A review of population-based optimization algorithms," in *2020 International Conference in Mathematics, Computer Engineering and Computer Science (ICMCECS)*, 2020, pp. 1–7.

[54] R. Cheng, C. He, Y. Jin, and X. Yao, "Model-based evolutionary algorithms a short survey," *Complex & Intelligent Systems*, vol. 4, no. 4, pp. 283–292, 2018.

[55] T. Sripramong and C. Toumazou, "The invention of cmos amplifiers using genetic programming and current-flow analysis," *IEEE transactions on computer-aided design of integrated circuits and systems*, vol. 21, no. 11, pp. 1237–1252, 2002.

[56] O. Mitea, M. Meissner, and L. Hedrich, "Topology synthesis of analog circuits with yield optimization and evaluation using pareto fronts," in *2011 IEEEIFIP 19th International Conference on VLSI and System-on-Chip*. IEEE, 2011, pp. 78–81.

[57] F. Jiao and A. Doboli, "A low-voltage, low-power amplifier created by reasoning-based, systematic topology synthesis," in *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2015, pp. 2648–2651.

[58] M. P.-H. Lin, Y.-W. Chang, and C.-M. Hung, "Recent research development and new challenges in analog layout synthesis," in *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*. IEEE, 2016, pp. 617–622.

[59] S. P. Boyd and S. J. Kim, "Geometric programming for circuit optimization," in *Proceedings of the 2005 international symposium on Physical design*, 2005, pp. 44–46.

[60] O. Okobiah, S. P. Mohanty, and E. Kougianos, "Exploring kriging for fast and accurate design optimization of nanoscale analog circuits," in *2014 IEEE Computer Society Annual Symposium on VLSI*. IEEE, 2014, pp. 244–247.

[61] O. Okobiah, S. Mohanty, and E. Kougianos, "Fast design optimization through simple kriging metamodeling a sense amplifier case study," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 4, pp. 932–937, 2013.

[62] T. McConaghy and G. Gielen, "Analysis of simulation-driven numerical performance modeling techniques for application to analog circuit optimization," in *2005 IEEE International Symposium on Circuits and Systems*. IEEE, 2005, pp. 1298–1301.

[63] R. A. Rutenbar, G. G. Gielen, and J. Roychowdhury, "Hierarchical modeling, optimization, and synthesis for system-level analog and rf designs," *Proceedings of the IEEE*, vol. 95, no. 3, pp. 640–669, 2007.

[64] M. del Mar Hershenson, "Design of pipeline analog-to-digital converters via geometric programming," in *Proceedings of the 2002 IEEEACM international conference on Computer-aided design*, 2002.

[65] J. Yuan, N. Farhat, and J. Van der Spiegel, "Gbopcad a synthesis tool for high-performance gain-boosted opamp design," *IEEE Transactions on Circuits and Systems I Regular Papers*, vol. 52, no. 8, 2005.

[66] S.-H. Lui, H.-K. Kwan, and N. Wong, "Analog circuit design by non-convex polynomial optimization two design examples," *International Journal of Circuit Theory and Applications*, vol. 38, no. 1, 2010.

[67] A. K. Singh, K. Ragab, M. Lok, C. Caramanis, and M. Orshansky, "Predictable equation-based analog optimization based on explicit capture of modeling error statistics," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, no. 10, 2012.

[68] T. McConaghy, P. Palmers, G. Gielen, and M. Steyaert, "Simultaneous multi-topology multi-objective sizing across thousands of analog circuit topologies," in *2007 44th ACMIEEE Design Automation Conference*. IEEE, 2007, pp. 944–947.

[69] S. Hammouda *et al.*, "Chameleon art a non-optimization based analog design migration framework," in *2007 44th ACMIEEE Design Automation Conference*, 2006, pp. 885–888.

[70] M. Dessouky, A. Kaiser, M.-M. Louerai, and A. Greiner, "Analog design for reuse-case study very low-voltagespl deltaspl sigmamodulator," in *Proceedings Design, Automation and Test in Europe. Conference and Exhibition 2001*. IEEE, 2001, pp. 353–360.

[71] R. Vural and T. Yildirim, "Analog circuit sizing via swarm intelligence," *AEU-International journal of electronics and communications*, vol. 66, no. 9, pp. 732–740, 2012.

[72] B. Liu *et al.*, "Analog circuit optimization system based on hybrid evolutionary algorithms," *Integration*, vol. 42, no. 2, pp. 137–148, 2009.

[73] G. Huang, L. Qian, S. Saibua, D. Zhou, and X. Zeng, "An efficient optimization based method to evaluate the drv of sram cells," *IEEE Transactions on Circuits and Systems I Regular Papers*, vol. 60, no. 6, pp. 1511–1520, 2013.

[74] A. Nieuwoudt and Y. Massoud, "Multi-level approach for integrated spiral inductor optimization," in *Proceedings of the 42nd annual Design Automation Conference*, 2005, pp. 648–651.

[75] B. Peng, F. Yang, C. Yan, X. Zeng, and D. Zhou, "Efficient multiple starting point optimization for automated analog circuit optimization via recycling simulation data," in *2016 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2016, pp. 1417–1422.

[76] M. Li, D. Zhou, S.-G. Wang, and X. Zeng, "Fmssqp an efficient global optimization tool for the robust design of rail-to-rail op-amp," in *2013 IEEE 10th International Conference on ASIC*. IEEE, 2013, pp. 1–4.

[77] G. Huang, D. Zhou, X. Zeng, and S. Wang, "A practical method for auto-design and optimization of dc-dc buck converter," in *2013 IEEE 10th International Conference on ASIC*. IEEE, 2013, pp. 1–4.

[78] L. Qian, Z. Bi, D. Zhou, and X. Zeng, "Automated technology migration methodology for mixed-signal circuit based on multistart optimization framework," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 11, pp. 2595–2605, 2014.

[79] M. Krasnicki, R. Phelps, R. A. Rutenbar, and L. R. Carley, "Maelstrom efficient simulation-based synthesis for custom analog cells," in *Proceedings 1999 Design Automation Conference (Cat. No. 99CH36361)*. IEEE, 1999, pp. 945–950.

[80] M. Naumowicz, M. Melosik, and P. Katarzy'nski, "Technology migration of analogue cmos circuits using hooke-jeeves algorithm and genetic algorithms in multi-core cpu systems," in *Proceedings of the 20th International Conference Mixed Design of Integrated Circuits and Systems-MIXDES 2013*. IEEE, 2013, pp. 267–272.

[81] D. Liu, S. Sidiropoulos, and M. Horowitz, "A framework for designing reusable analog circuits," in *ICCAD-2003. International Conference on Computer Aided Design (IEEE Cat. No. 03CH37486)*. IEEE, 2003.

[82] G. İslamoğlu, T. O. Çakici, E. Afacan, and G. Dündar, "Artificial neural network assisted analog ic sizing tool," in *2019 16th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*. IEEE, 2019, pp. 9–12.

[83] Y. Li, Y. Wang, Y. Li, R. Zhou, and Z. Lin, "An artificial neural network assisted optimization system for analog design space exploration," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 10, pp. 2640–2653, 2019.

[84] B. Liu, D. Zhao, P. Reynaert, and G. G. Gielen, "Synthesis of integrated passive components for high-frequency rf ics based on evolutionary computation and machine learning techniques," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 10, pp. 1458–1468, 2011.

[85] B. Liu, Y. He, P. Reynaert, and G. Gielen, "Global optimization of integrated transformers for high frequency microwave circuits using a gaussian process based surrogate model," in *2011 Design, Automation & Test in Europe*. IEEE, 2011, pp. 1–6.

[86] B. Liu, N. Deferm, D. Zhao, P. Reynaert, and G. G. Gielen, "An efficient high-frequency linear rf amplifier synthesis method based on evolutionary computation and machine learning techniques," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, no. 7, pp. 981–993, 2012.

[87] B. Liu, D. Zhao, P. Reynaert, and G. G. Gielen, "Gaspad a general and efficient mm-wave integrated circuit synthesis method based on surrogate model assisted evolutionary algorithm," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 2, pp. 169–182, 2014.

[88] D. M. Colleran *et al.*, "Optimization of phase-locked loop circuits via geometric programming," in *Proceedings of the IEEE 2003 Custom Integrated Circuits Conference, 2003.* IEEE, 2003, pp. 377–380.

[89] W. Daems, G. Gielen, and W. Sansen, "Simulation-based generation of posynomial performance models for the sizing of analog integrated circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 22, no. 5, pp. 517–534, 2003.

[90] Y. Wang, M. Orshansky, and C. Caramanis, "Enabling efficient analog synthesis by coupling sparse regression and polynomial optimization," in *Proceedings of the 51st Annual Design Automation Conference*, 2014, pp. 1–6.

[91] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *Journal of Global optimization*, vol. 13, no. 4, pp. 455–492, 1998.

[92] B. Liu and A. Nikolaeva, "Efficient global optimization of mems based on surrogate model assisted evolutionary algorithm," in *2016 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2016, pp. 555–558.

[93] S. J. Park, B. Bae, J. Kim, and M. Swaminathan, "Application of machine learning for optimization of 3-d integrated circuits and systems," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 6, pp. 1856–1865, 2017.

[94] 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," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2015.

[95] C. E. Rasmussen, "Gaussian processes in machine learning," in *Summer School on Machine Learning*. Springer, 2003, pp. 63–71.

[96] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006, vol. 2, no. 3.

[97] M. Schonlau, W. J. Welch, and D. R. Jones, "Global versus local search in constrained optimization of computer models," *Lecture Notes-Monograph Series*, pp. 11–25, 1998.

[98] J. R. Gardner, M. J. Kusner, Z. E. Xu, K. Q. Weinberger, and J. P. Cunningham, "Bayesian optimization with inequality constraints." in *ICML*, vol. 2014, 2014, pp. 937–945.

[99] R. T. Marler and J. S. Arora, "Survey of multi-objective optimization methods for engineering," *Structural and multidisciplinary optimization*, vol. 26, no. 6, pp. 369–395, 2004.

[100] C. Chevalier and D. Ginsbourger, "Fast computation of the multi-points expected improvement with applications in batch selection," in *International Conference on Learning and Intelligent Optimization*. Springer, 2013, pp. 59–69.

[101] J. Wu and P. Frazier, "The parallel knowledge gradient method for batch bayesian optimization," in *Advances in Neural Information Processing Systems*, 2016, pp. 3126–3134.

[102] A. Shah and Z. Ghahramani, "Parallel predictive entropy search for batch global optimization of expensive objective functions," in *Advances in Neural Information Processing Systems*, 2015.

[103] B. Liu, F. V. Fernández, and G. G. Gielen, "Efficient and accurate statistical analog yield optimization and variation-aware circuit sizing based on computational intelligence techniques," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 6, pp. 793–805, 2011.

[104] A. Singhee, S. Singhal, and R. A. Rutenbar, "Practical, fast monte carlo statistical static timing analysis: why and how," in *2008 IEEE/ACM International Conference on Computer-Aided Design*. IEEE, 2008.

[105] M. Sengupta, S. Saxena, L. Daldoss, G. Kramer, S. Minehane, and J. Cheng, "Application-specific worst case corners using response surfaces and statistical models," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 9, 2005.

[106] M. Wang *et al.*, "Efficient yield optimization for analog and sram circuits via gaussian process regression and adaptive yield estimation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 10, pp. 1929–1942, 2018.

[107] J. Nocedal and S. Wright, *Numerical optimization*. Springer Science & Business Media, 2006.

[108] Y. Yang *et al.*, "Smart-msp a self-adaptive multiple starting point optimization approach for analog circuit synthesis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 3, pp. 531–544, 2017.

[109] F. Glover, "Tabu search—part i," *ORSA Journal on computing*, vol. 1, no. 3, pp. 190–206, 1989.

[110] F. Glover, "Tabu search—part ii," *ORSA Journal on computing*, vol. 2, no. 1, pp. 4–32, 1990.

[111] G. Alpaydin, S. Balkir, and G. Dundar, "An evolutionary approach to automatic synthesis of high-performance analog integrated circuits," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 3, 2003.

[112] G. Wolfe and R. Vemuri, "Extraction and use of neural network models in automated synthesis of operational amplifiers," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 22, no. 2, pp. 198–212, 2003.

[113] H. Liu, A. Singhee, R. A. Rutenbar, and L. R. Carley, "Remembrance of circuits past macromodeling by data mining in large analog design spaces," in *Proceedings of the 39th annual Design Automation Conference*, 2002, pp. 437–442.

[114] C. Cortes, M. Mohri, and A. Rostamizadeh, "L2 regularization for learning kernels," *arXiv preprint arXiv:1205.2653*, 2012.

[115] P. Liashchynskyi and P. Liashchynskyi, "Grid search, random search, genetic algorithm: A big comparison for nas," *arXiv preprint arXiv:1912.06059*, 2019.

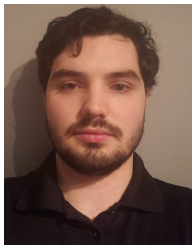
[116] P.-C. Pan, C.-C. Huang, and H.-M. Chen, "Late breaking results an efficient learning-based approach for performance exploration on analog and rf circuit synthesis," in *2019 56th ACMIEEE Design Automation Conference (DAC)*. IEEE, 2019, pp. 1–2.

[117] K.-H. Meng, P.-C. Pan, and H.-M. Chen, "Integrated hierarchical synthesis of analog rf circuits with accurate performance mapping," in *2011 12th International Symposium on Quality Electronic Design*. IEEE, 2011, pp. 1–8.

[118] P.-C. Pan, H.-M. Chen, and C.-C. Lin, "Page parallel agile genetic exploration towards utmost performance for analog circuit design," in *Proceedings of the Conference on Design, Automation and Test in Europe*, 2013, pp. 1849–1854.

[119] E. Chang *et al.*, "Bag2 a process-portable framework for generator-based ams circuit design," in *2018 IEEE Custom Integrated Circuits Conference (CICC)*. IEEE, 2018, pp. 1–8.
[120] J. Snoek *et al.*, "Scalable bayesian optimization using deep neural networks," in *International conference on machine learning*, 2015.
[121] W. Huang, D. Zhao, F. Sun, H. Liu, and E. Chang, "Scalable gaussian process regression using deep neural networks," in *Twenty-fourth international joint conference on artificial intelligence*. Citeseer, 2015.
[122] Z. Wang, K. H. Lee, and N. Verma, "Overcoming computational errors in sensing platforms through embedded machine-learning kernels," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 8, pp. 1459–1470, 2014.
[123] J. González, Z. Dai, P. Hennig, and N. Lawrence, "Batch bayesian optimization via local penalization," in *Artificial intelligence and statistics*. PMLR, 2016, pp. 648–657.
[124] T. Desautels, A. Krause, and J. W. Burdick, "Parallelizing exploration-exploitation tradeoffs in gaussian process bandit optimization," *Journal of Machine Learning Research*, vol. 15, pp. 3873–3923, 2014.
[125] M. Wang, F. Yang, C. Yan, X. Zeng, and X. Hu, "Efficient bayesian yield optimization approach for analog and sram circuits," in *2017 54th ACMEDACIEEE Design Automation Conference (DAC)*. IEEE, 2017.
[126] F. Wang, M. Zaheer, X. Li, J.-O. Plouchart, and A. Valdes-Garcia, "Co-learning bayesian model fusion efficient performance modeling of analog and mixed-signal circuits using side information," in *2015 IEEEACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2015, pp. 575–582.
[127] X. Li and H. Liu, "Statistical regression for efficient high-dimensional modeling of analog and mixed-signal performance variations," in *Proceedings of the 45th annual Design Automation Conference*, 2008.
[128] X. Li, "Finding deterministic solution from underdetermined equation large-scale performance modeling by least angle regression," in *2009 46th ACMIEEE Design Automation Conference*. IEEE, 2009.
[129] W. Zhang, T.-H. Chen, M.-Y. Ting, and X. Li, "Toward efficient large-scale performance modeling of integrated circuits via multi-modemulti-corner sparse regression," in *Design Automation Conference*. IEEE, 2010, pp. 897–902.
[130] X. Li, "Finding deterministic solution from underdetermined equation large-scale performance variability modeling of analogrf circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 29, no. 11, pp. 1661–1668, 2010.
[131] X. Li, W. Zhang, and F. Wang, "Large-scale statistical performance modeling of analog and mixed-signal circuits," in *Proceedings of the IEEE 2012 Custom Integrated Circuits Conference*. IEEE, 2012.
[132] T. McConaghy, "High-dimensional statistical modeling and analysis of custom integrated circuits," in *2011 IEEE Custom Integrated Circuits Conference (CICC)*. IEEE, 2011, pp. 1–8.
[133] W. Zhang *et al.*, "Virtual probe a statistical framework for low-cost silicon characterization of nanoscale integrated circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 12, pp. 1814–1827, 2011.
[134] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.
[135] A. E. Hoerl and R. W. Kennard, "Ridge regression biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, 1970.
[136] X. Li, W. Zhang, F. Wang, S. Sun, and C. Gu, "Efficient parametric yield estimation of analogmixed-signal circuits via bayesian model fusion," in *2012 IEEEACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2012, pp. 627–634.
[137] C. Fang, F. Yang, X. Zeng, and X. Li, "Bmf-bd bayesian model fusion on bernoulli distribution for efficient yield estimation of integrated circuits," in *2014 51st ACMEDACIEEE Design Automation Conference (DAC)*. IEEE, 2014, pp. 1–6.
[138] F. Wang *et al.*, "Bayesian model fusion large-scale performance modeling of analog and mixed-signal circuits by reusing early-stage data," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 8, pp. 1255–1268, 2015.
[139] Q. Huang, C. Fang, F. Yang, X. Zeng, and X. Li, "Efficient multivariate moment estimation via bayesian model fusion for analog and mixed-signal circuits," in *Proceedings of the 52nd Annual Design Automation Conference*, 2015, pp. 1–6.
[140] C. Fang *et al.*, "Efficient bit error rate estimation for high-speed link by bayesian model fusion," in *2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2015.
[141] M. Alawieh, F. Wang, and X. Li, "Efficient hierarchical performance modeling for integrated circuits via bayesian co-learning," in *Proceedings of the 54th Annual Design Automation Conference 2017*, 2017.
[142] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.
[143] S. Yu, B. Krishnapuram, H. Steck, R. B. Rao, and R. Rosales, "Bayesian co-training," in *Advances in neural information processing systems*, 2008, pp. 1665–1672.
[144] H. Ishwaran, J. S. Rao *et al.*, "Spike and slab variable selection frequentist and bayesian strategies," *The Annals of Statistics*, vol. 33, no. 2, pp. 730–773, 2005.
[145] G. Casella and E. I. George, "Explaining the gibbs sampler," *The American Statistician*, vol. 46, no. 3, pp. 167–174, 1992.
[146] G.-H. Liou, S.-H. Wang, Y.-Y. Su, and M. P.-H. Lin, "Classifying analog and digital circuits with machine learning techniques toward mixed-signal design automation," in *2018 15th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*. IEEE, 2018, pp. 173–176.
[147] S. M. Bowers, K. Sengupta, K. Dasgupta, and A. Hajimiri, "A fully-integrated self-healing power amplifier," in *2012 IEEE Radio Frequency Integrated Circuits Symposium*. IEEE, 2012, pp. 221–224.

Morteza Fayazi was born in Tehran, Iran, in 1994. He received the B.Sc. major degree in electrical engineering and minor degree in computer science from Sharif University of Technology (SUT), Tehran, Iran, in 2017. He Also received the M.S. degree in electrical engineering and computer science in 2020 from University of Michigan, Ann Arbor, MI. He is currently working toward the Ph.D. degree at University of Michigan, Ann Arbor, MI. His research interests include circuit design automation, machine learning, software development and computer architecture.



Zachary Colter was born in Michigan, United States in 1997. He recieved both the B.Sc. degree, in computer engineering, and M.S. degree, in electrical computer engineering, from the University of Michigan. The degrees were acquired in 2019 and 2020 respectively.

Ehsan Afshari was born in 1979. He received the B.Sc. degree in electronics engineering from Sharif University of Technology, Tehran, Iran, and the M.S. and Ph.D. degrees in electrical engineering from the California Institute of Technology, Pasadena, in 2003 and 2006, respectively. In August 2006, he joined the faculty in Electrical and Computer Engineering at Cornell University, Ithaca, NY. His research interests are high-speed and low-noise integrated circuits for applications in communication systems, sensing, and biomedical devices. Prof. Afshari serves as the chair of the IEEE Ithaca section, as the chair of Cornell Highly Integrated Physical Systems (CHIPS), as a member of the Analog Signal Processing Technical Committee of the IEEE Circuits and Systems Society, and a member of the Technical Program Committee of the IEEE Custom Integrated Circuits Conference.

Ronald Dreslinski received the BSE degrees both in electrical engineering and computer engineering, and the MSE and PhD degrees in computer science and engineering from the University of Michigan, Ann Arbor, Michigan. He is currently an assistant professor of computer science and engineering with the University of Michigan. His research focuses on architectures that enable emerging low-power circuit techniques.