Analog Circuit Yield Optimization via Freeze–Thaw Bayesian Optimization Technique

Xiaodong Wang, Changhao Yan, *Member, IEEE*, Yuzhe Ma, *Member, IEEE*, Bei Yu, *Member, IEEE*, Fan Yang, *Member, IEEE*, Dian Zhou, *Senior Member, IEEE*, and Xuan Zeng, *Senior Member, IEEE*

***Abstract*—While the VLSI community cares about designs with high yields under process variations, expensive computational costs make conventional yield optimization methods for analog circuits inefficient for industrial applications. In this article, an efficient yield optimization method via the freeze–thaw Bayesian optimization technique is proposed for analog circuits. The yield analysis is integrated into the exploration process of the Bayesian optimization. With a specified Gaussian process regression method, the flexible freeze–thaw Bayesian optimization technique is utilized to automatically guide the search in the design space and control the accuracy of yield analysis in the process space. A performance optimization problem is formulated and solved to mine prior knowledge, and a further speed up is achieved. Experimental results show that the proposed method can gain a 2.47×–5.73× speedup compared with the state-of-the-art methods, without loss of accuracy.**

***Index Terms*—Design for space exploration, freeze–thaw Bayesian optimization, transistor sizing, yield modeling, yield optimization.**

I. Introduction

As SEMICONDUCTOR fabrication technology scales to the nanometer level, process variations have strong impacts on the performances, yields of analog circuits. In order to deal with the increasing challenges in reliable circuit designs, the IC community pays more and more attention to yield optimization recently [1], [2].

In general, yield optimization flow features an iterative loop as designers first adjusting design parameters, like sizes of transistors, then executing the time-consuming yield analysis. Since yield analysis (PVT or Monte Carlo simulations) needs thousands of simulations to guarantee the accuracy, the time cost of the entire yield optimization is extremely high. Considering the narrowing time-to-market, reducing the overall simulation time of yield optimization for analog circuits is the most urgent requirement.

To resolve this issue, a large number of methodologies and algorithms have been proposed, mainly including the following three categories. Corner-based methods [3]–[7] optimize the "worst case" performance of given circuits at several process corners. Although this treatment avoids costly yield estimations, it is coarse, inaccurate, and often leads to overdesign. In addition, it is difficult to search the worst case in a high-dimensional process space.

Monte-Carlo (MC)-based methods are the most straightforward and widely used methods due to their high accuracy and generality. Liu *et al.* [8] and Guerra-Gomez *et al.* [9] applied the optimal computation budget allocation (OCBA) techniques for MC speedup, and evolutionary algorithms for optimization. Wang *et al.* [10] employed the kernel density estimation method for yield modeling and proposed a multistart-point expectation–maximization-like algorithm to solve the problem. Wang *et al.* [11] proposed an adaptive yield analysis method and implemented Bayesian optimization with the weighted expected improvement (wEI) criterion to obtain the optimal design. Zhang *et al.* [12] employed the Gaussian process (GP) regression with the neural network and max-value entropy search (ES) methods for optimization based on [11]. Although Bayesian optimization has shown certain advantages in previous research, the computational costs are still a major deterrent to mainstream adoption for yield optimization. For example, the state-of-the-art methods [11], [12] need 6000–20 000 simulations, which is time prohibitive for analog circuits.

Response-surface-based methods [13]–[16] try to build a surrogate model of circuit performance to replace the expensive simulator, thus reducing the cost of yield optimization. However, these methods are limited for requiring lots of samples to maintain modeling accuracy and being nearly impossible to build the surrogate model in a high-dimensional space.

As we only care about the optimal design, the simulation resources allocated to selected designs should be dynamically adjusted. In fact, the idea of applying coarse yield estimations for low-yield designs has been successfully introduced in [11] for saving simulations, as shown in the low-yield region in Fig. 1(a), where the x-axis is the number of samples, the y-axis is the yield, and different lines are different designs. Those designs whose yields are much lower than design 1 are discarded with cheap estimations, e.g., design 4 and design 5. However, for designs with yields close to design 1, e.g.,

Manuscript received 21 June 2021; revised 12 September 2021 and 14 December 2021; accepted 14 January 2022. Date of publication 7 February 2022; date of current version 24 October 2022. This work was supported in part by the National Key R&D Program of China under Grant 2020YFA0711900 and Grant 2020YFA0711901; in part by the National Natural Science Foundation of China (NSFC) Research Projects under Grant 62141407, Grant 61974032, Grant 61822402, Grant 62090025, and Grant 61929102; and in part by the Research Grants Council of Hong Kong SAR under Grant CUHK14209420. This article was recommended by Associate Editor S. Mohanty. *(Corresponding authors: Changhao Yan; Xuan Zeng.)*

Xiaodong Wang, Changhao Yan, Fan Yang, and Xuan Zeng are with the State Key Laboratory of ASIC & System, Microelectronics Department, Fudan University, Shanghai 200433, China (e-mail: yanch@fudan.edu.cn; xzeng@fudan.edu.cn).

Yuzhe Ma is with Microelectronics Thrust, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou 511458, China.

Bei Yu is with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, SAR.

Dian Zhou is with the Department of Electrical Engineering, The University of Texas at Dallas, Richardson, TX 75080 USA.

Digital Object Identifier 10.1109/TCAD.2022.3149723

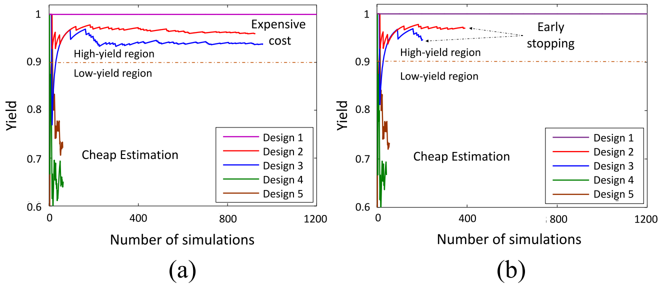


Fig. 1. Comparison of (a) traditional adaptive analysis and (b) freeze–thaw technique for yield optimization.

design 2 and design 3, [11] cannot distinguish them effectively, and has to calculate their high-accuracy yields immediately, leading to huge simulation cost. Intuitively, if we collect designs with potentially high yields and gradually improve their analysis accuracy rather than execute high-accuracy yield analysis at once, simulation cost may be reduced because yield analysis at most designs can be early stopped when a higher-yield region is found, even if they were ever considered candidates for optimal design in the early stage of the optimization procedure.

Recently, curve learning-related modeling methods have aroused widespread interest in the machine learning community [17], [18]. Inspired by [17], a freeze–thaw Bayesian optimization technique is introduced in this article for yield optimization. The basic idea of *freeze–thaw* is to integrate yield analysis into the exploration process of Bayesian optimization, automatically guide the search in the design space and allocate computational resources of yield estimations. By modeling the process of yield analysis, we can temporarily *freeze* the analysis at one design, if its predicted final yield does not seem to be optimal. We can also thaw the analysis if it becomes the optimal candidate again. Even we can start analysis at a new design.

Concretely, improving the yield accuracy of a design is considered as an iterative process throughout the entire optimization, i.e., samples are gradually added to improve the analysis accuracy. The freeze–thaw Bayesian optimization puts and freezes the high-yield designs in a candidate basket for their yields are currently blurry, predicts the next design with optimal yield and selects the design from basket, which maximally reduces the uncertainty of the optimal (i.e., information gain), to execute yield analysis. This technique gradually improves the accuracy of yield analysis in the high-yield region, and most designs will be allocated only a small number of simulations as shown in Fig. 1(b).

Empirically, a good design should have a high yield and good circuit performances, i.e., satisfying performance specifications, which hints us to intuitively find the design with optimal yield near those designs with good performances. Compared with the expensive yield optimization, the performance optimization at the typical–typical (TT) corner is much cheaper. Therefore, we innovatively embed the relatively cheap performance optimization into the yield optimization flow. The performance optimization at the beginning of yield optimization is able to provide a good start for yield optimization, resulting in an efficient *warm start*, which shows considerable advantage on efficiently exploring the large optimization space.

In this article, a general and efficient yield optimization method for analog circuits is proposed. Some key contributions are concluded as follows.

1) A freeze–thaw Bayesian optimization technique is first applied for the yield optimization of analog circuits, which automatically guides the search in the design space and gradually improves the analysis accuracy in the process space. With the high sampling efficiency of Bayesian optimization and flexibility of the freeze–thaw technique, the overall costs of yield optimization are significantly reduced.
2) A nominal performance optimization problem is formulated and solved in the yield optimization framework. This treatment helps to mine prior knowledge for yield optimization, and a further speedup is achieved.
3) Experimental results show that the proposed method achieves a 2.47×–5.73× speedup over the state-of-the-art methods, without loss of accuracy.

The remainder of this article is organized as follows. In Section II, we present the problem formulation and traditional methods for yield estimation, modeling, and optimization. The complete yield optimization approach via a freeze–thaw technique is introduced in Section III. The implementation details are discussed in Section IV. Experimental results are given to validate the proposed method in Section V. Finally, Section VI concludes this article.

II. Background

A. Problem Formulation

In design space $D \subseteq R^{d_x}$, a design point $\mathbf{x} = [x_1, x_2, \ldots, x_{d_x}]^\top \in D$ means a d_x-dimensional vector. Design parameters $x_i, i = 1, \ldots, d_x$ are restricted in reasonable ranges $[l_i, u_i]$, respectively, representing values of bias voltages and currents, widths and lengths of transistors, etc.

In process space $V \subseteq R^{d_s}$, a process point $\mathbf{s} = [s_1, s_2, \ldots, s_{d_s}]^\top \in V$ denotes a d_s-dimensional vector. Process parameters $s_i, i = 1, \ldots, d_s$ are random variables modeling the variations of manufacturing process, such as threshold voltages, etc., following normal distributions provided by foundries [19].

Generally, in most process design kits (PDKs), process parameters are independent of the design parameters and mutually independent [10]. The probability density function (PDF) of $\mathbf{s}$ is

$$p(\mathbf{s}) = \prod_{i=1}^{d_s} \left[\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{s_i^2}{2} \right) \right]. \tag{1}$$

Circuit performances $\mathbf{y} = [y_1, y_2, \ldots, y_k]^\top$ can be regarded as a function of design and process parameters: $y_i = f_{\text{sim}}^i(\mathbf{x}, \mathbf{s}), \mathbf{x} \in D, \mathbf{s} \in V$, where f_{sim}^i means that the ith performance y_i can be obtained from a transistor-level simulation f_{sim}. Usually, circuits are supposed to meet corresponding specifications $\mathbf{c} = [c_1, c_2, \ldots, c_k]^\top$ set by designers. A circuit is *success* if all specifications are satisfied, i.e., $y_i \geq c_i, i = 1, \ldots, k$ without loss of generality. Otherwise, it is a *fail*.

Given design parameters $\mathbf{x}$, the yield $Y(\mathbf{x})$ can be expressed as

$$Y(\mathbf{x}) = \int_V I(\mathbf{x}, \mathbf{s}) p(\mathbf{s}) d\mathbf{s} \tag{2}$$

where indicator function $I(\mathbf{x}, \mathbf{s}) = \text{AND}(y_i \geq c_i), i = 1, \dots, k$, and $\text{AND}(\cdot)$ is logical function *AND*.

The goal of the yield optimization problem is to find a design point $\mathbf{x}^*$ with maximal yield Y as

$$\mathbf{x}^* = \arg\max_{\mathbf{x} \in D} \quad Y(\mathbf{x}). \tag{3}$$

B. Monte Carlo Analysis

MC methods are the most widely used numerical methods for probability density analysis. For a given design $\mathbf{x}$, with N random samples $\mathbf{s}_i, i = 1, \dots, N$ drawn from $p(\mathbf{s})$, MC estimates the yield $Y(\mathbf{x})$ as

$$\widehat{Y}_{\text{MC}}(\mathbf{x}) = \frac{1}{N} \sum_{i=1}^{N} I(\mathbf{x}, \mathbf{s}_i). \tag{4}$$

The variance of the estimator $\widehat{Y}_{\text{MC}}(\mathbf{x})$ is determined by the sampling size N as [20]

$$\sigma_{\widehat{Y}}^2 \approx \frac{\widehat{Y}_{\text{MC}}\big(1 - \widehat{Y}_{\text{MC}}\big)}{N} \cdot k_\gamma^2 \tag{5}$$

where the confidence level k_γ is a constant. For example, $k_\gamma = 1.645$ corresponds to a 90% confidence level.

By effectively utilizing the relationship between confidence interval $\sigma_{\widehat{Y}}^2$ and sample size N, the number of MC samples required for one yield estimation can be adjusted by given target yield and specific accuracy.

C. Gaussian Process Regression Method

The GP regression model [21] is a powerful and effective nonparametric probabilistic method, which has been widely studied in many engineering fields, due to its ability to provide both posterior mean and corresponding uncertainty. Given a finite collection of n points, $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$, $\mathbf{x}_i \in D, i = 1, \dots, n$, and the unknown true yield $\mathbf{Y} = \{Y(\mathbf{x}_i), i = 1, \dots, n\}$, GP is defined as a probability distribution over $\mathbf{Y}$ such that $\mathbf{Y} = \{Y(\mathbf{x}_i), i = 1, \dots, n\}$ jointly have a multivariate Gaussian distribution

$$\mathbf{Y} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K}) \tag{6}$$

where $\boldsymbol{\mu}$ is an $n \times 1$ mean vector, and $\mathbf{K}$ is an $n \times n$ covariance matrix.

GP is fully specified by its prior mean function $m(\mathbf{x})$ and its covariance function $k(\mathbf{x}_i, \mathbf{x}_j)$. In (6), the mean vector $\boldsymbol{\mu}$ is determined by $m(\mathbf{x})$, i.e., $\mu_i = m(\mathbf{x}_i), i = 1, \dots, n$, and $k(\mathbf{x}_i, \mathbf{x}_j)$ determines the covariance matrix $\mathbf{K}$, i.e., $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j), i, j = 1, \dots, n$. Usually, $m(\mathbf{x})$ is set to a constant for convenience. As for the covariance function, there are usually multiple options, e.g., the squared exponential kernel and Matérn kernels. In this article, Matérn-5/2 kernel is selected as

$$k\big(\mathbf{x}_i, \mathbf{x}_j\big) = \sigma_f^2\left(1 + \sqrt{5}r + \frac{5}{3}r^2\right)\exp\left(-\sqrt{5}r\right), \; i, j = 1 \cdots n \tag{7}$$

where $r^2 = (\mathbf{x}_i - \mathbf{x}_j)^\top \boldsymbol{\Lambda}^{-1}(\mathbf{x}_i - \mathbf{x}_j)$. σ_f denotes the *output variance*, and $\boldsymbol{\Lambda} = \text{diag}({l_1}^2, \dots, {l_n}^2)$, $l_i, i = 1, \dots, n$ represents the ith *characteristic length scale*.

Considering the noise $\epsilon_x \sim \mathcal{N}(0, \sigma_x^2)$ of statistical yield analysis, noise-corrupted estimated yield $\widehat{\mathbf{Y}} = \{\widehat{Y}(\mathbf{x_i}), i = 1, \dots, n\}$ has the form

$$\widehat{\mathbf{Y}} \sim \mathcal{N}\Big(\mathbf{Y}, \sigma_x^2 \mathbf{I}\Big) \tag{8}$$

where $\mathbf{I}$ is the identity matrix. Then, the covariance function for the elements of matrix $\mathbf{K}'$ becomes

$$k'\big(\mathbf{x}_i, \mathbf{x}_j\big) = k\big(\mathbf{x}_i, \mathbf{x}_j\big) + \sigma_x^2 \delta_{ij} \tag{9}$$

where $\delta_{ij} = 1$ if $i = j$, otherwise $\delta_{ij} = 0$.

The hyperparameters σ_f, $l_i, i = 1, \dots, n$ and σ_x in the GP model can be determined by the maximum likelihood estimation (MLE). Given a new design $\mathbf{x}_a$, its corresponding yield $\widehat{Y}(\mathbf{x}_a)$ and observed yields $\mathbf{Y}$ follow the joint Gaussian distribution:

$$\begin{bmatrix} \mathbf{Y} \\ \widehat{Y}(\mathbf{x}_a) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu} \\ m(\mathbf{x}_a) \end{bmatrix}, \begin{bmatrix} \mathbf{K}' & \mathbf{k}'(\mathbf{x}_a, \mathbf{X})^\top \\ \mathbf{k}'(\mathbf{x}_a, \mathbf{X}) & k'(\mathbf{x}_a, \mathbf{x}_a) \end{bmatrix}\right) \tag{10}$$

where $\mathbf{k}'(\mathbf{x}_a, \mathbf{X}) = [k'(\mathbf{x}_a, \mathbf{x}_1), k'(\mathbf{x}_a, \mathbf{x}_2), \dots, k'(\mathbf{x}_a, \mathbf{x}_n)]$.

Then, the predictive distribution of $\widehat{Y}(\mathbf{x}_a)$ conditioned on observations $\{\mathbf{X}, \mathbf{Y}\}$ can be derived with the following posterior mean and variance:

$$\mu\big(\widehat{Y}(\mathbf{x}_a)\big) = \mathbf{k}'(\mathbf{x}_a, \mathbf{X})^\top \mathbf{K}'^{-1}\mathbf{Y} \tag{11}$$

$$\sigma^2\big(\widehat{Y}(\mathbf{x}_a)\big) = k'(\mathbf{x}_a, \mathbf{x}_a) - \mathbf{k}'(\mathbf{x}_a, \mathbf{X})^\top \mathbf{K}'^{-1}\mathbf{k}'(\mathbf{x}_a, \mathbf{X}). \tag{12}$$

D. Bayesian Optimization

Since yield analysis tools are extremely costly and the estimation results are always noise corrupted, Bayesian optimization [22]–[24] has attracted widespread attention recently in the yield optimization problem [11], [12], [25]. In general, the Bayesian optimization framework has two key components. The first component is a probabilistic model, which reflects our beliefs about the unknown objective function, e.g., the GP model in Section II-C. The second component is an acquisition function that utilizes the posterior distribution to guide the search. By maximizing the acquisition function, Bayesian optimization aims to balance the tradeoff between exploitation and exploration, i.e., the next query point is located where the model prediction is high (exploitation) and/or the model uncertainty is large (exploration). Specifically, there are several kinds of acquisition functions, e.g., traditional Thompson sampling (TS) [26], lower confidence bound (LCB) [27], knowledge gradient [28], and the newly proposed acquisition function, named probability of further improvement (PFI) which aims to deal with stringent specifications for analog IC sizing problems [29].

A typical acquisition function is the expected improvement (EI) [30], which has a closed form. Given the current maximum yield τ, the improvement of $Y(\mathbf{x})$ predicted by GP model can be calculated with $\text{I}(Y(\mathbf{x})) = \max(0, Y(\mathbf{x}) - \tau)$. Since $Y(\mathbf{x})$ follows a Gaussian distribution, EI is expressed as:

$$\text{EI}(\mathbf{x}) = \mathbb{E}[\text{I}(Y(\mathbf{x}))] = \sigma(\mathbf{x})(\lambda\Phi(\lambda) + \phi(\lambda)) \tag{13}$$

where $\lambda = (\tau - \mu(\mathbf{x}))/\sigma(\mathbf{x})$, $\mu(\mathbf{x})$ and $\sigma(\mathbf{x})$ are the posterior mean and standard deviation of $Y(\mathbf{x})$ in (11) and (12). $\Phi(\cdot)$ and $\phi(\cdot)$ represent the CDF and PDF of the standard normal distribution, respectively.

For the yield optimization problem, nominal performances can be set as constraints [11], [25], i.e., all specifications should be satisfied at the TT corner. This treatment can help to reduce unfeasible design space for speedup. The constrained yield optimization problem is written as

$$\begin{aligned}\mathbf{x}^* = \arg\max_{\mathbf{x}\in D} \quad & Y(\mathbf{x}) \\ \text{s.t.} \quad & y_i(\mathbf{x}, \text{TT}) \geq c_i, \; i = 1, \ldots, k \end{aligned} \tag{14}$$

where $y_i(\mathbf{x}, \text{TT})$ is the ith nominal performance of design $\mathbf{x}$, and $\mathbf{x}^*$ is the global optimal design.

It should be noted that other design costs (e.g., area and power, etc.), besides the performances, can be further integrated into this formulation. We can either add them in the objective function with weight coefficients to optimize them simultaneously [31], or set them as additional constraints to meet specific requirements [32].

To deal with the constraints, the wEI [33] is proposed to multiply the EI function with the probability of feasibility (PF), i.e., the probability of all constraints being satisfied, written as

$$\text{wEI}(\mathbf{x}) = \text{PF}(\mathbf{x}) \cdot \text{EI}(\mathbf{x}) \tag{15}$$

where $\text{PF}(\mathbf{x}) = \prod_{i=1}^{k} P(y_i(\mathbf{x}, \text{TT}) \geq c_i)$.

Different from *wEI*, which focuses on the change of yield value, another acquisition function adopted in this article, i.e., ES [34], considers more about the location of the optimal design. Formally, it can be expressed as

$$\text{ES}(\mathbf{x}) = \int \left(H(P_{\mathbf{x}^*}) - H\left(P^{Y}_{\mathbf{x}^*}\right)\right) P(Y|D_o) dY \tag{16}$$

where the observed data $D_o = \{\mathbf{X}, \mathbf{Y}\}$. $\mathbf{x}^*$ represents the unknown design with the optimal yield. $H(\cdot)$ denotes the differential entropy, and D_o' is the updated data set of D_o with one more data point $(\mathbf{x}_{n+1}, \widehat{Y}(\mathbf{x}_{n+1}))$ observed. $P_{\mathbf{x}^*}$ and $P^{Y}_{\mathbf{x}^*}$ are the estimated distributions over $\mathbf{x}^*$ given observed data D_o and D_o', respectively, [17].

Fig. 4(a) illustrates the ordinary Bayesian optimization flow. At each iteration, the probabilistic model is built with current training data. Then, the acquisition function can be computed with the predictive posterior distribution, and it is optimized to generate the next query design, e.g., $\mathbf{x}^*_{\text{wEI}}$. After estimating yield at $\mathbf{x}^*_{\text{wEI}}$, the data set is updated with this new sample. These steps are taken iteratively until the termination condition is reached.

III. Proposed Approach

This section will present the proposed yield optimization method for analog circuits. First, a freeze–thaw GP regression method for iterative yield analysis is discussed. Then, the framework of freeze–thaw Bayesian optimization technique is described. Next, a nominal performance optimization problem is formulated and solved to mine prior knowledge. Finally, a complete flow summarizes the proposed method.

A. Freeze–Thaw Gaussian Process Regression Model for Iterative Yield Analysis

In general, the yield analysis needs to be invoked repetitively during the yield optimization process, thus reducing analysis costs is an important way to improve the algorithm efficiency. In previous yield optimization methods [8], [9], [11], [12], one design will only be selected once, the obtained yield value will not change after estimation. However, the proposed approach allows the same design to be repeatedly selected for estimation during the optimization. Specifically, when a design is selected for the first time, we will execute a rough analysis at it. If it is picked again later, we will add more samples to improve its analysis accuracy. Intuitively, the design with the highest yield will be repeatedly selected until the termination condition is reached.

In other words, the yield analysis is integrated into the exploration process of Bayesian optimization. Yield analysis at any given design is treated as an iterative process rather than executed at once. From this perspective, the overall optimization process is regarded as a double loop, where the outer loop explores the design space and selects candidate designs, the inner loop adds samples for yield estimations. Since this analysis method is MC-based, the estimated yield will gradually converge to its true value with samples added.

To model this iterative analysis process, a freeze–thaw GP regression method is designed. With the ability to predict the final yield with partially completed analysis, the computational costs of yield estimations can be significantly reduced.

Formally, given designs $\{\mathbf{x}_i\}_{i=1}^{n}$, let $g_i^j, j = 1, \ldots, T_i$ denote estimated yield value with j batches of simulations sampled at $\mathbf{x}_i$. $T_i \in \mathbb{N}^+$ indicates the total number of batches sampled at $\mathbf{x}_i$. $\mathbf{g}_i = [g_i^1, g_i^2, \ldots, g_i^{T_i}]$ is a T_i-dimensional vector representing the analysis curve, and $\mathbf{t}_i = [1, 2, \ldots, T_i]$ is the corresponding time steps of $\mathbf{g}_i$. In order to build a surrogate model for every analysis curve, a specified kernel [17] is utilized. For $t_i^a, t_i^b \in \mathbf{t}_i$, the kernel function $k(t_i^a, t_i^b)$ is given by

$$k\left(t_i^a, t_i^b\right) = \frac{\beta^\alpha}{\left(t_i^a + t_i^b + \beta\right)^\alpha} \tag{17}$$

where α and β are two hyperparameters. A feature of this kernel function is that its value will tend to be constant when t_i^a or t_i^b is large, so it is suitable for modeling a curve which gradually converges.

In this article, the specified kernel is used as the covariance function over time steps for an iterative yield analysis curve. Concretely, each curve is drawn from a separate GP, modeled by

$$\mathcal{N}\left(\mathbf{g}_i; Y_i \mathbf{1}_i, \mathbf{K}_{\mathbf{t}_i\mathbf{t}_i}\right) \tag{18}$$

where Y_i is the asymptotic value that the ith curve converges to, i.e., final yield at $\mathbf{x}_i$, and $\mathbf{1}$ is a column vector of 1's. The specified curve kernel is selected for the elements of $\mathbf{K}_{\mathbf{t}_i\mathbf{t}_i}$, $i = 1, \ldots, n$.

Considering the correlation of yields in the design space, the final yield Y_i is regarded as a latent function that specifies the asymptotic value of each analysis curve, jointly modeled by

$$\mathcal{N}(\mathbf{Y}; \mathbf{m}, \mathbf{K}_{\mathbf{xx}}) \tag{19}$$

where $\mathbf{Y} = \{Y_1, Y_2, \ldots, Y_n\}$. The prior mean $\mathbf{m} = [m_1, m_2, \ldots, m_n]$ remains constant, and m_i is the mean of $\mathbf{g}_i$, $i = 1, \ldots, n$. Matérn-5/2 kernel is selected for the elements of $\mathbf{K}_{\mathbf{xx}}$.

Subsequently, the distribution over all analysis curves $\{\mathbf{g}_i\}_{i=1}^{n}$ can be written as

$$P\left(\{\mathbf{g}_i\}_{i=1}^{n} | \{\mathbf{x}_i\}_{i=1}^{n}\right) = \int \left[\prod_{i=1}^{n} \mathcal{N}\left(\mathbf{g}_i; Y_i \mathbf{1}_i, \mathbf{K}_{\mathbf{t}_i\mathbf{t}_i}\right)\right] \cdot \mathcal{N}(\mathbf{Y}; \mathbf{m}, \mathbf{K}_{\mathbf{xx}}) d\mathbf{Y}. \tag{20}$$

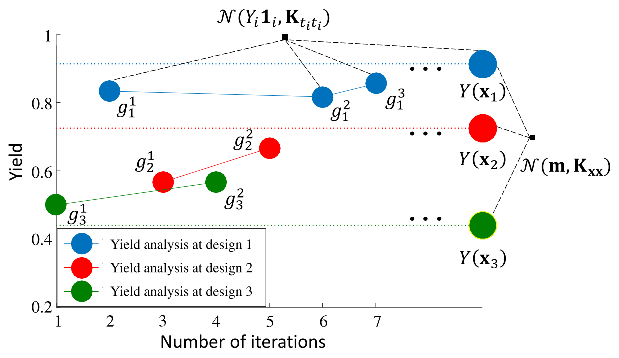

$\mathcal{N}(Y_i\mathbf{1}_i, \mathbf{K}_{t_it_i})$
$\mathcal{N}(\mathbf{m}, \mathbf{K}_{\mathbf{xx}})$
Yield
1
0.8
0.6
0.4
0.2
g_1^1
g_1^2
g_1^3
g_2^1
g_2^2
g_3^1
g_3^2
$Y(\mathbf{x}_1)$
$Y(\mathbf{x}_2)$
$Y(\mathbf{x}_3)$
Yield analysis at design 1
Yield analysis at design 2
Yield analysis at design 3
1
2
3
4
5
6
7
Number of iterations

Fig. 2. Illustration of the GP model for iterative yield analysis.

Fig. 2 illustrates the basic idea of this GP model where x-axis is the number of iterations, y-axis is the yield. Each line represents an yield analysis curve $\mathbf{g}_i$ at a given design $\mathbf{x}_i$. It can be seen that the analysis procedure is executed as an iterative process throughout the entire optimization rather than at once. Each curve is modeled by $\mathcal{N}(Y_i\mathbf{1}_i, \mathbf{K}_{\mathbf{t}_i\mathbf{t}_i})$, $i = 1, \ldots, n$ and the final yields Y_i, $i = 1, \ldots, n$ are jointly modeled by $\mathcal{N}(\mathbf{m}, \mathbf{K}_{\mathbf{xx}})$. In other words, the final yields Y_i, $i = 1, \ldots, n$ are first drawn over design space according to a GP prior. Conditioned on Y_i, each analysis curve is modeled independently using another GP prior.

As yield estimations are always noise corrupted, the statistical noise $\epsilon_t \sim \mathcal{N}(0, \sigma_t^2)$ is counted. Then, the covariance function for the elements of matrix $\mathbf{K'}_{\mathbf{t}_i\mathbf{t}_i}$ becomes

$$k'\left(t_i^a, t_i^b\right) = k\left(t_i^a, t_i^b\right) + \sigma_t^2\delta_{ab} \tag{21}$$

where $\delta_{ab} = 1$ if $a = b$; otherwise, $\delta_{ab} = 0$. To obtain the hyperparameters, the log likelihood is derived from (20) as

$$\begin{aligned}\log P\left(\widehat{\mathbf{g}}|\{\mathbf{x}_i\}_{i=1}^n\right) = &-\frac{1}{2}(\widehat{\mathbf{g}} - \mathbf{Om})^\top \mathbf{K'}_{\mathbf{tt}}^{-1}(\widehat{\mathbf{g}} - \mathbf{Om})\\ &+ \frac{1}{2}\boldsymbol{\gamma}^\top\left(\mathbf{K'}_{\mathbf{xx}}^{-1} + \mathbf{Q}\right)^{-1}\boldsymbol{\gamma}\\ &- \frac{1}{2}\left(\log\left(|\mathbf{K'}_{\mathbf{xx}}^{-1} + \mathbf{Q}|\right)\right.\\ &+ \log\left(|\mathbf{K'}_{\mathbf{xx}}|\right) + \log\left(|\mathbf{K'}_{\mathbf{tt}}|\right) + \text{const}\end{aligned} \tag{22}$$

where $\widehat{\mathbf{g}} = [\widehat{\mathbf{g}}_1, \widehat{\mathbf{g}}_2, \ldots, \widehat{\mathbf{g}}_n]^\top$, represents the vector composed of all yield analysis curves together. $\mathbf{O} = \text{blockdiag}(\mathbf{1}_1, \mathbf{1}_2, \ldots, \mathbf{1}_n)$ is a block-diagonal matrix. Its block $\mathbf{1}_i$ is a vector of ones with length T_i, $i = 1, \ldots, n$. $\mathbf{K'}_{\mathbf{tt}} = \text{blockdiag}(\mathbf{K'}_{\mathbf{t}_1\mathbf{t}_1}, \mathbf{K'}_{\mathbf{t}_2\mathbf{t}_2}, \ldots, \mathbf{K'}_{\mathbf{t}_n\mathbf{t}_n})$ is a block-diagonal matrix with blocks $\mathbf{K'}_{\mathbf{t}_i\mathbf{t}_i}$. $\boldsymbol{\gamma}$ is an n-dimensional vector with elements $\gamma_i = \mathbf{1}^\top\mathbf{K'}_{\mathbf{t}_i\mathbf{t}_i}^{-1}(\widehat{\mathbf{g}}_i - m_i)$, and $\mathbf{Q} = \text{diag}(q_1, \ldots, q_n)$ with elements $q_i = \mathbf{1}^\top\mathbf{K'}_{\mathbf{t}_i\mathbf{t}_i}^{-1}\mathbf{1}$, $i = 1, \ldots, n$.

All hyperparameters, including α, β, σ_t, etc., can be learned by MLE.

Using the Bayesian inference, we can derive the required quantities for optimization from (22). Formally, given all observed designs $\{\mathbf{x}_i\}_{i=1}^n$ and noise-corrupted analysis curves $\{\widehat{\mathbf{g}}_i\}_{i=1}^n$, the posterior distribution of final yields $\widehat{\mathbf{Y}}$ at old designs $\{\mathbf{x}_i\}_{i=1}^n$ is expressed as

$$\begin{aligned}p\left(\widehat{\mathbf{Y}}|\{\widehat{\mathbf{g}}_i\}_{i=1}^n, \{\mathbf{x}_i\}_{i=1}^n\right) &= \mathcal{N}\left(\widehat{\mathbf{Y}}; \boldsymbol{\mu}, \mathbf{C}\right)\\ \boldsymbol{\mu} &= \mathbf{m} + \boldsymbol{C}\boldsymbol{\gamma}\\ \mathbf{C} &= \mathbf{K'}_{\mathbf{xx}} - \mathbf{K'}_{\mathbf{xx}}\left(\mathbf{K'}_{\mathbf{xx}} + \mathbf{Q}^{-1}\right)^{-1}\mathbf{K'}_{\mathbf{xx}}.\end{aligned} \tag{23}$$

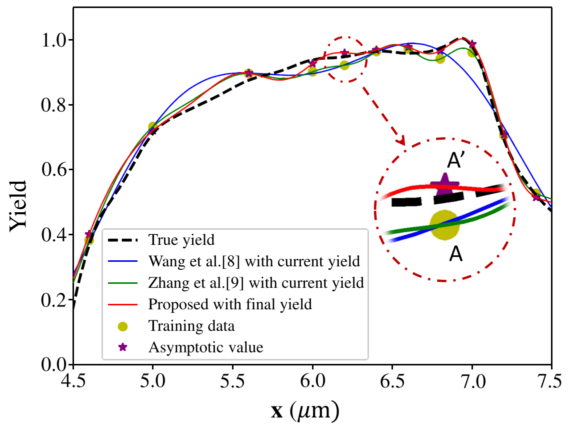


Fig. 3. Yield regression comparison between the proposed method and existing methods in the design space.

Then, for a new design $\mathbf{x}_a$, the posterior mean $\mu(\widehat{Y}(\mathbf{x}_a))$ and uncertainty measurement $\sigma^2(\widehat{Y}(\mathbf{x}_a))$ of its final yield $\widehat{Y}(\mathbf{x}_a)$ are written as

$$\mu\left(\widehat{Y}(\mathbf{x}_a)\right) = \mathbf{k'}_{\mathbf{xx}_a}^\top\mathbf{K'}_{\mathbf{xx}}^{-1}\boldsymbol{\mu} \tag{24}$$

$$\sigma^2\left(\widehat{Y}(\mathbf{x}_a)\right) = k'_{\mathbf{x}_a\mathbf{x}_a} - \mathbf{k'}_{\mathbf{xx}_a}^\top\left(\mathbf{K'}_{\mathbf{xx}} + \mathbf{Q}^{-1}\right)^{-1}\mathbf{k'}_{\mathbf{xx}_a}. \tag{25}$$

Fig. 3 shows the yield regression comparison in design space between the proposed method and existing methods [11], [12] under the same training data. The x-axis is the width of transistor M19 in a comparator circuit (Fig. 9) and the y-axis is the yield. As the yield data are often roughly estimated, yellow dots (estimated yield) may not accurately fall on the black-dotted line (true yield). Existing methods show poor fitting results since they take the noisy current yields as golden solutions, i.e., they try to fit the yellow dots, e.g., point A. However, the proposed method first predicts the final yields of current designs, e.g., point A', and then utilizes them for further yield prediction of new designs. Equations (11) and (24) show this difference intuitively, where (24) uses the final yields $\boldsymbol{\mu}$ for better prediction, rather than the current yields $\mathbf{Y}$ in (11). It can be clearly seen that the predicted final yield (point A') is closer to the true yield (black-dotted line) than current yield (point A) at the same design. Thus, the proposed method can provide better regression accuracy.

B. Framework of Freeze–Thaw Bayesian Optimization Technique

To reduce the simulation cost of the yield optimization problem, a freeze–thaw Bayesian optimization method is proposed as follows. Fig. 4(b) shows the optimization flow. First, a basket B is constructed to collect candidates with potentially high yields during the optimization. Specifically, the criterion for selecting candidates is to choose the top N_B designs with the current highest lower bound of the estimated yield, formally written as

$$B = \left\{\arg\max_{1:N_B}\left(\widehat{Y}_{\text{MC}}(\mathbf{x}) - \sigma_{\widehat{Y}}(\mathbf{x})\right), \mathbf{x} \in \mathbf{X}\right\} \tag{26}$$

where $\widehat{Y}_{\text{MC}}(\mathbf{x})$ and $\sigma_{\widehat{Y}}(\mathbf{x})$ are calculated with (4) and (5). Since the yield analysis may be rough at the early and mid term, maintaining a certain number of candidates that have already been estimated to some degree can avoid missing the optimal design. The basket size N_B is chosen to be 10 in this article.

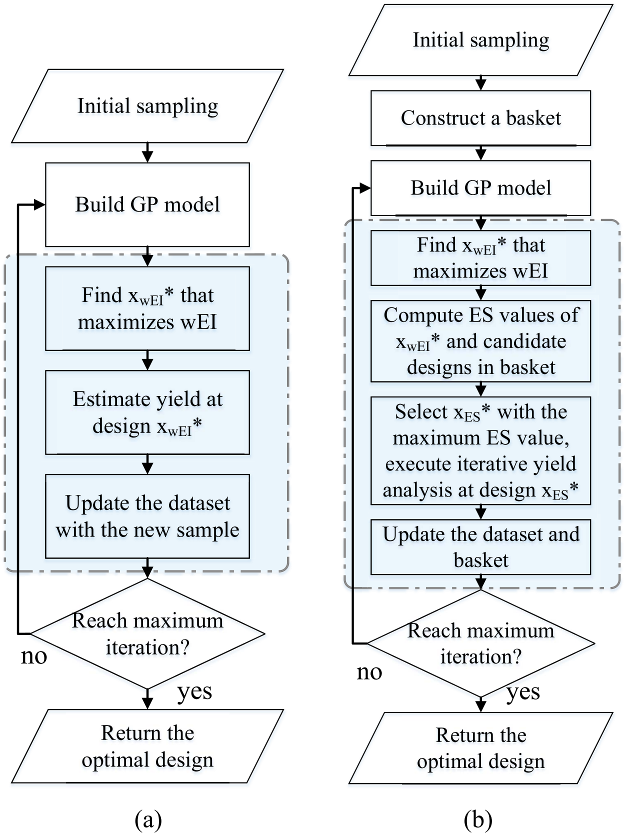

Initial sampling
Build GP model
Find x_wEI* that maximizes wEI
Estimate yield at design x_wEI*
Update the dataset with the new sample
Reach maximum iteration?
no
yes
Return the optimal design
(a)
Initial sampling
Construct a basket
Build GP model
Find x_wEI* that maximizes wEI
Compute ES values of x_wEI* and candidate designs in basket
Select x_ES* with the maximum ES value, execute iterative yield analysis at design x_ES*
Update the dataset and basket
Reach maximum iteration?
no
yes
Return the optimal design
(b)

Fig. 4. Comparison between (a) ordinary Bayesian optimization and (b) freeze–thaw Bayesian optimization.

Then the freeze–thaw GP model described in Section III-A is built. Different from previous methods, which use only one acquisition function to determine the next query point, either [11] with wEI or [12] with ES, as shown in Fig. 4(a), the proposed method effectively leverages two acquisition functions to guide the search in design space, shown in Fig. 4(b). It has been reported in [35] that different acquisition functions may lead to conflicting results. Therefore, this treatment could make the results of design selection more comprehensive.

Concretely, after model training, the wEI maximization problem in (27) is solved by a multiple starting point (MSP) strategy to obtain a new design $\mathbf{x}^*_{\text{wEI}}$ in each iteration as

$$\mathbf{x}^*_{\text{wEI}} = \arg\max_{\mathbf{x}\in D} \quad \text{wEI}(\mathbf{x}). \tag{27}$$

It has been shown in [36]–[38] that MSP strategy is very effective for global optimization. The BFGS method [39] is chosen for local search under the MSP framework.

Next, we compare the ES values of $\mathbf{x}^*_{\text{wEI}}$ and candidate designs in the basket. ES is usually approximated with MC methods [17], [40]. The implementation details for ES in freeze–thaw method will be further discussed in Section IV-A. The design with the highest ES value is selected as the next query point, denoted as $\mathbf{x}^*_{\text{ES}}$. This step can be formulated as

$$\mathbf{x}^*_{\text{ES}} = \arg\max_{\mathbf{x}\in B\cup\mathbf{x}^*_{\text{wEI}}} \quad \text{ES}(\mathbf{x}). \tag{28}$$

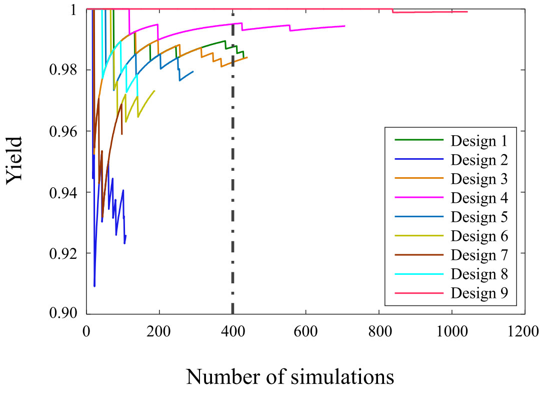


Fig. 5. Analysis curves in one yield optimization execution of a comparator circuit.

Subsequently, one batch of simulations will be sampled at $\mathbf{x}^*_{\text{ES}}$ to estimate its yield. Specifically, if $\mathbf{x}^*_{\text{ES}} = \mathbf{x}^*_{\text{wEI}}$, i.e., the new design $\mathbf{x}^*_{\text{wEI}}$ is the most promising point, the current estimation process in the basket will be paused (freeze), and yield analysis at a brand-new design will be executed with one batch of simulations. Otherwise, a previous partially completed estimation at an old design in the basket will continue (thaw) with one more batch of samples. By this two-round selection, the sampling efficiency in terms of yield estimations can be further improved.

After that, the training data set is updated with the newly sampled observation, and the basket is rebuilt with the top N_B designs with the current highest lower bound of the estimated yield of all training data, using (26). More competitive designs will be added into the basket, while some old candidate designs may be removed, even including the optimal designs ever found in the early and mid term. The optimization procedure continues until a user-defined maximal number of iterations is reached.

Essentially, the basket is a tradeoff between two acquisition functions, i.e., wEI and ES to determine the next query design. Concretely, if the basket size, i.e., the number of candidates maintained, is very small, e.g., zero, the algorithm tends to select $\mathbf{x}^*_{\text{wEI}}$ for yield analysis. This means it will always execute analysis on new designs. On the other hand, if the basket size becomes infinite, obtaining $\mathbf{x}^*_{\text{ES}}$ is equivalent to searching for optimal ES value in the whole design space. $\mathbf{x}^*_{\text{wEI}}$ is almost impossible to be selected as $\mathbf{x}^*_{\text{ES}}$, then this method will always execute analysis on old designs. Thus, maintaining a moderate number of candidates is important for the freeze–thaw technique, so that the algorithm could switch analysis between new designs and old designs according to the acquisition functions, wEI and ES.

Fig. 5 shows the analysis curves with yields higher than 90% in one optimization execution of a comparator circuit (see Fig. 9) where the x-axis is the number of simulations and the y-axis is the yield. Clearly, different designs are allocated different amounts of simulations.

Actually, the adaptive yield analysis has been introduced in [11] and [12]. However, they allocate simulation resources based on the comparison with current maximum yield τ. If the yield of the estimated design is close to τ, a large number of simulations will be sampled. However, τ gradually increases during the entire optimization process; thus, this treatment may lead to a waste of simulation resources, in the case that the algorithm has not yet explored the optimal-yield region.

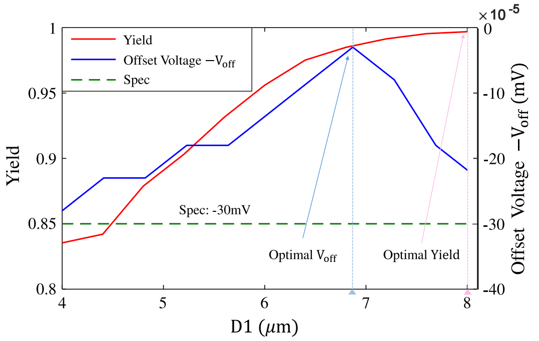


Fig. 6. Yield and nominal performance of a comparator circuit.

Specifically, suppose there is a design with true yield Y_s and an optimal design with true yield τ. In order to confirm which one is better in [11] and [12], according to (5), the number of samples required for MC analysis can be calculated with

$$N \approx \frac{Y_s(1-Y_s)}{(\tau - Y_s)^2} \cdot k_\gamma^2. \tag{29}$$

Take $Y_s = 94\%$ for example, it will be allocated over 1000 simulations when τ is 95%. However, when τ increases to 98%, less than 100 points are sufficient. The proposed method uses a basket to collect candidates with potentially high yields, instead of comparing their yields immediately, allowing the designs found in the early stage to compete with those found in the later stage for resources, automatically cutting down the simulation costs.

For the case in Section V-A, many designs can reach a yield value of 97%, but only four designs with yields greater than 97% are allocated more than 400 simulations as shown in Fig. 5.

C. Knowledge Transfer From Nominal Design

In the freeze–thaw Bayesian optimization, the basket is built with random initialization. In order to further reduce the simulation cost, we need to pick some prior solutions for the basket. Actually, the cost of circuit performance optimization is much lower than that of yield optimization. For example, only a few hundred simulations are needed in a performance optimization execution [32], [41], [42]. However, even a single yield analysis may cost thousands of simulations. Naturally, we consider using the results of performance optimization to guide the yield optimization.

Fig. 6 shows the yield and nominal performance of a comparator circuit, i.e., the first case in the experimental section, where the D1-axis represents the width of transistor M17. The performance metric is the offset voltage V_{off} with 200-MHz sampling frequency at 30 °C, and the specification is $V_{off} \leq$ 30mV. We can see that a design with the best performance does not usually have the optimal yield. On the contrary, those designs which roughly meet and are close to the specifications, i.e., near the failure boundary, may have better yields.

Based on this observation, we formulate a nominal performance optimization problem as

$$\min_{\mathbf{x}\in D} \; f(\mathbf{x}) = \sum_{i=1}^{k} \omega_i \cdot |(y_i(\mathbf{x}, \mathrm{TT}) - \epsilon \cdot c_i)/c_i|$$
$$\text{s.t.} \;\; y_i(\mathbf{x}, \mathrm{TT}) \geq c_i, \;\; i = 1, \ldots, k \tag{30}$$

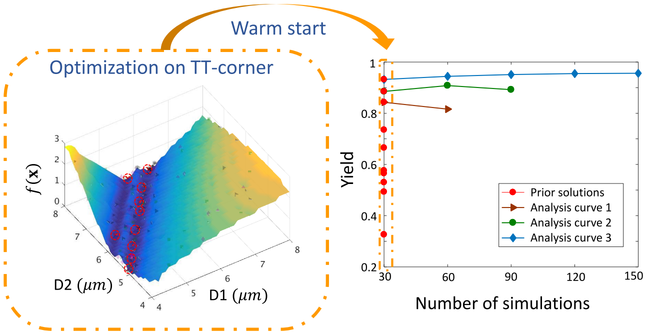


Fig. 7. Prior knowledge transferred to yield optimization.

where ω_i is the weight, representing the importance of the ith performance metric. ϵ is a constant coefficient, determining the desired distance between the nominal performance value $y_i(\mathbf{x}, \mathrm{TT})$ and specification c_i. We set $\omega_i = 1, i = 1, \ldots, k$ and $\epsilon = (3/2)$ empirically in experiments.

In this article, the nominal performance optimization problem is solved by the approach proposed in [32], i.e., WEIBO, with a maximum iteration number of N_{pre}, which is a state-of-the-art performance optimization method for analog circuits. To make the obtained prior solutions more widely distributed, this solver is invoked three times randomly and independently, resulting in $3 \times N_{\mathrm{pre}}$ simulation costs. Then, all the obtained designs meeting the constraints will be divided into N clusters by the k-means method. Every design with the smallest $f(\mathbf{x})$ in corresponding cluster is selected as the prior solution. A total number of N designs are added to the data set and basket for yield optimization.

Fig. 7 shows the proposed strategy to leverage the knowledge from nominal performance optimization to reduce the cost of yield optimization. In the left part, D1-axis and D2-axis are the widths of transistor M17 and M19, respectively, in a comparator circuit (see Fig. 9). The z-axis is the optimization target $f(\mathbf{x})$ in (30), and its origin corresponds to the position where nominal performances $y_i(\mathbf{x}, \mathrm{TT})$ are equal to $\epsilon \cdot c_i, i = 1, \ldots, n$. The points sampled in the three optimizations on TT-corner are marked with diamonds, triangles, and squares, and the selected prior solutions are marked with red circles. In the right part, the x-axis is the number of simulations and the y-axis is the yield. Several prior solutions get high yields, leading to an efficient warm start. By picking prior solutions for the basket at the initial stage, prior knowledge is transferred to yield optimization. Concretely, the prior knowledge helps to reduce the number of simulations by approximately 20% on average. Moreover, this strategy is particularly useful in the problems exploring large optimization space. For example, in Section V-E, existing methods [11], [12] may fail to obtain the optimal design, but the proposed method succeeds by leveraging the prior knowledge.

D. Summary

The proposed yield optimization flow for analog circuits is depicted in Fig. 8. It consists of two parts. One is the nominal performance optimization part used to mine prior knowledge, as described in Section III-C. It is worth mentioning that our proposed approach is orthogonal to the solver for nominal performance optimization. The other is the yield optimization part via the freeze–thaw technique. Throughout the optimization process, yield analysis at one design is

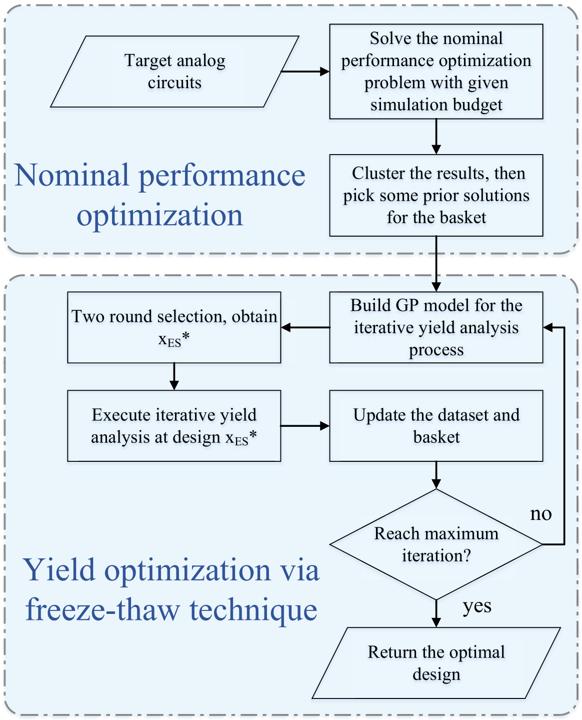

Target analog circuits
Solve the nominal performance optimization problem with given simulation budget
Nominal performance optimization
Cluster the results, then pick some prior solutions for the basket
Two round selection, obtain x_{ES}*
Build GP model for the iterative yield analysis process
Execute iterative yield analysis at design x_{ES}*
Update the dataset and basket
Reach maximum iteration?
no
yes
Yield optimization via freeze-thaw technique
Return the optimal design

Fig. 8. Proposed yield optimization flow for analog circuits.

executed iteratively. The analysis accuracy can be gradually improved by the freeze–thaw strategy, as described in Section III-B.

IV. Implementation Details

A. Entropy Search for Freeze–Thaw Technique

ES considers the uncertainty reduction over the location of the optimal design when new observations, $(\mathbf{x}_{\text{fant}}, Y_{\text{fant}})$ are added, and iteratively evaluates designs which will most improve the information gain. Since $(\mathbf{x}_{\text{fant}}, Y_{\text{fant}})$ is not really evaluated, ES is often calculated with *fantasized* yields predicted by the GP model [40]. Concretely, by adding the fantasized observations $(\mathbf{x}_{\text{fant}}, Y_{\text{fant}})$ to the training set, ES selects the design that causes the maximum uncertainty reduction.

Due to the iterative yield analysis, yield values are gradually updated during the optimization. Whether a brand-new design is estimated for the first time, i.e., $(\mathbf{x}_{n+1}, \widehat{g}^1_{n+1})$, or an old design is taken one step further, i.e., $(\mathbf{x}_i, \widehat{g}^{T_i+1}_i)$, $i \in \{1, \ldots, n\}$, new information about the optimal location will be provided. To compute the information gain, we have to calculate the fantasized yields of new designs, i.e., $\mathbf{x}^*_{\text{wEI}}$ and old designs, i.e., $\mathbf{x} \in B$ with one more batch of simulations added.

Formally, based on (20), the yield value of an old design in the basket if one more batch of simulations are added can be derived, e.g., the $(T+1)$th point on the ith curve, $\mathbf{t}'_i = [T+1]$. The posterior mean and variance of $\widehat{\mathbf{g}}^{T+1}_i$ are given by

$$\begin{cases} \mu\left(\widehat{\mathbf{g}}^{T+1}_i\right) = \mathbf{k'}^{\top}_{\mathbf{t}_i\mathbf{t}'_i}\mathbf{K'}^{-1}_{\mathbf{t}_i\mathbf{t}_i}\widehat{\mathbf{g}}_i + \Omega\mu_i \\ \sigma^2\left(\widehat{\mathbf{g}}^{T+1}_i\right) = k'_{\mathbf{t}'_i\mathbf{t}'_i} - \mathbf{k'}^{\top}_{\mathbf{t}_i\mathbf{t}'_i}\mathbf{K'}^{-1}_{\mathbf{t}_i\mathbf{t}_i}\mathbf{k'}_{\mathbf{t}_i\mathbf{t}'_i} + C_{ii}\Omega^2 \end{cases} \tag{31}$$

Algorithm 1 Calculation of Differential Entropy

```
Require: A trained freeze-thaw GP model.
1: Select $N_d$ representer points with the highest wEI values using a Markov chain Monte Carlo sampler;
2: Draw $N_f$ samples from the freeze-thaw GP model;
3: Calculate the probability of each representer design becoming optimal, obtain $P_{\mathbf{x}^*}$;
4: Calculate the differential entropy value $H(P_{\mathbf{x}^*})$.
```

where $\Omega = 1 - \mathbf{k'}^{\top}_{\mathbf{t}_i\mathbf{t}'_i}\mathbf{K'}^{-1}_{\mathbf{t}_i\mathbf{t}_i}\mathbf{1}$, μ_i is the ith element of $\boldsymbol{\mu}$, and C_{ii} is the ith diagonal element of $\mathbf{C}$. Specifically, $\mathbf{k'}_{\mathbf{t}_i\mathbf{t}'_i} = [k'(1, T+1), k'(2, T+1), \ldots, k'(T, T+1)]^{\top}, i = 1, \ldots, n$.

As for $\mathbf{x}^*_{\text{wEI}}$, its yield with one batch of simulations sampled can also be derived. For the first point on the $(n+1)$th curve at $\mathbf{x}^*_{\text{wEI}}$, $\mathbf{t}_{n+1} = [1]$, the posterior distribution of $\widehat{\mathbf{g}}^1_{n+1}$ is written as

$$\begin{aligned} &P\left(\widehat{\mathbf{g}}^1_{n+1} | \{\widehat{\mathbf{g}}_i\}^n_{i=1}, \{\mathbf{x}_i\}^n_{i=1}, \mathbf{x}_{n+1}\right) \\ &\quad = \mathcal{N}\left(\widehat{\mathbf{g}}^1_{n+1}; \mu\left(\widehat{Y}(\mathbf{x}_{n+1})\right), \sigma^2\left(\widehat{Y}(\mathbf{x}_{n+1})\right) + \mathbf{k'}_{\mathbf{t}_{n+1}\mathbf{t}_{n+1}}\right) \end{aligned} \tag{32}$$

where $\mu(\widehat{Y}(\mathbf{x}_{n+1}))$ and $\sigma^2(\widehat{Y}(\mathbf{x}_{n+1}))$ are calculated with (24) and (25), respectively.

With these information provided by the freeze–thaw GP model, we can calculate the probability of the optimal design, $P_{\mathbf{x}^*}$ and its differential entropy value $H(P_{\mathbf{x}^*})$ [17], [40]. Algorithm 1 describes the procedure of calculating them using MC estimation in detail.

First, we select some points named representer points in the design space which should be sensitive to the probability change of the unknown optimal design $\mathbf{x}^*$. Instead of random sampling, we use a Markov chain Monte Carlo sampler to find representers with the target measure set as the wEI function for it tending to have high value in high-yield regions where $P_{\mathbf{x}^*}$ is also large. The set of representers R can be written as

$$R = \left\{\arg\max_{1:N_d}(\text{wEI}), \mathbf{x} \in \mathbf{X}\right\}. \tag{33}$$

Then, we draw N_f samples from the posterior distribution provided by the freeze-thaw GP model. Each sample corresponds to a set of predicted yields of representer points. Next, we calculate the probability of each representer becoming optimal, resulting in an N_d-dimensional vector $P_{\mathbf{x}^*}$. Thus, the differential entropy value can be obtained by

$$H(P_{\mathbf{x}^*}) = -P_{\mathbf{x}^*} \cdot \log(P_{\mathbf{x}^*}). \tag{34}$$

In this article, N_d and N_f are set to 50 and 500, respectively.

Finally, ES for the freeze-thaw technique is described in Algorithm 2. In each iteration, after obtaining $\mathbf{x}^*_{\text{wEI}}$, $H(P_{\mathbf{x}^*})$ is first computed with currently observed data set D_o. Then, we calculate the fantasized yields of $\mathbf{x}_i \in B, i = 1, \ldots, N_B$ and $\mathbf{x}^*_{\text{wEI}}$ with (31) and (32). These fantasized observations are added to the training set, respectively. Denote $D_o{}'$ as the updated data set with fantasized observations added. The posterior distribution of the freeze–thaw GP model will change accordingly, leading to different $H(P^Y_{\mathbf{x}^*})$ and ES value. The design which maximizes the expected information gain over the optimal design is selected as $\mathbf{x}^*_{\text{ES}}$.

B. Computational Complexity

The computational cost for the training of conventional GP models is $\mathcal{O}(N^3)$ due to the inversion of the covariance matrix,

Algorithm 2 ES for Freeze–Thaw Technique

```
Require: Candidate designs, including old designs in the basket $B = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_{N_B}\}$ and a new design $\mathbf{x}^*_{wEI}$.
Initialize $\mathbf{ES} = [0, 0, \dots, 0]$ to represent information gain;
Compute $H(P_{\mathbf{x}^*})$ with current data using Monte Carlo estimation;
for $i = 1, 2, \dots, N_B + 1$ do
    if $\mathbf{x}_i \in B$ then
        Calculate a fantasized yield value $\widehat{\mathbf{g}}_i^{T+1}$ with (31);
    else
        Calculate a fantasized yield value $\widehat{\mathbf{g}}_i^{1}$ with (32);
    end if
    Add the fantasized observation to the data set, compute $H(P^Y_{\mathbf{x}^*})$ using Monte Carlo estimation;
    $\mathbf{ES}(i) \leftarrow H(P_{\mathbf{x}^*}) - H(P^Y_{\mathbf{x}^*})$;
end for
Select $\mathbf{x}^*_{ES}$ that maximizes $\mathbf{ES}$ as the next query point.
```

where N is the total number of training data, i.e., designs explored in [11] and the same as the model used in [21]. As for the freeze–thaw GP model applied in this article, since each design corresponds to an analysis curve, there are actually $N \cdot \overline{T}$ batches of samples, leading to a training data size of $N \cdot \overline{T}$, where N is the number of designs, and $\overline{T}$ represents the average number of batches per design. Outwardly, the computational complexity of the freeze–thaw model is prohibitively expensive as $\mathcal{O}(N^3\overline{T}^3)$. However, a careful derivation shows that its complexity is affordable indeed.

To train the GP model by MLE, we need to calculate the likelihood function. According to (22), there are four items that need the inversion of the covariance matrix, including $\mathbf{K}'^{-1}_{\mathbf{tt}}$, $\mathbf{K}'^{-1}_{\mathbf{xx}}$, $\boldsymbol{\gamma}$, and $\mathbf{Q}$. As $\mathbf{K}'_{\mathbf{tt}}$ is a block-diagonal matrix, we just need to compute the inversion of its blocks, leading to a complexity of $\mathcal{O}(N\overline{T}^3)$. Then, we save the Cholesky decomposition results of $\mathbf{K}'_{\mathbf{tt}}$ to calculate $\boldsymbol{\gamma}$ and $\mathbf{Q}$, corresponding to a complexity of $\mathcal{O}(N\overline{T}^2)$. In total, the computational complexity of this model is $\mathcal{O}(N^3 + N\overline{T}^3 + N\overline{T}^2)$, where $\mathcal{O}(N^3)$ comes from the computation of $\mathbf{K}'^{-1}_{\mathbf{xx}}$.

Intuitively, the reason for this complexity is the conditional independence assumption described in Section III-A, i.e., each analysis curve is drawn from an independent GP prior conditioned on the final yields drawn from a global GP. Analysis curves are modeled independently by $\mathcal{N}(Y_i\mathbf{1}_i, \mathbf{K}_{\mathbf{t}_i\mathbf{t}_i})$, $i = 1, \dots, n$ and their asymptotic values are jointly modeled by $\mathcal{N}(\mathbf{m}, \mathbf{K}'_{\mathbf{xx}})$.

In the experiments, simulations are conducted in batch fashion and each batch contains 30 runs for the iterative yield analysis procedure. The maximum number of simulations allocated to one design is 1200, so the maximum length $T_{max} = 40$. Due to the freeze–thaw technique, most analysis curves have only short lengths. The average length is $\overline{T} = 4$ in our experiments, which is much smaller than N, as the average value of N equals 141. If $\overline{T} \ll N$ is given, the computational complexity becomes $\mathcal{O}(N^3)$, which is comparable with the conventional GP models.

V. Experimental Results

In this section, the efficiency and efficacy of the proposed yield optimization approach will be demonstrated with four analog circuits: 1) comparator; 2) low noise amplifier; 3) three-stage amplifier; and 4) charge pump. We compare our method with two state-of-the-art methods [11], [12]. To average out the random fluctuations, all cases are executed ten times. Then, the yield of the obtained design $\mathbf{x}^*$ is estimated with 50 000 simulations in the process space to ensure the accuracy of yield analysis. Method FTBO represents the proposed method via freeze–thaw Bayesian optimization without prior knowledge. FTBO+ is the FTBO with prior knowledge transferred from nominal design. The total simulation budget of precomputation, i.e., nominal performance optimization, for FTBO+ is set to 300 ($N_{\text{pre}} = 100$) in Sections V-A–V-D. Considering the large design space, the budget in Section V-E is increased to 1800 ($N_{\text{pre}} = 600$). This precomputation cost has already been counted to the sampling number of FTBO+. All experiments are conducted on a Linux workstation with two Intel Xeon X5650 CPUs and 128-GB memory.

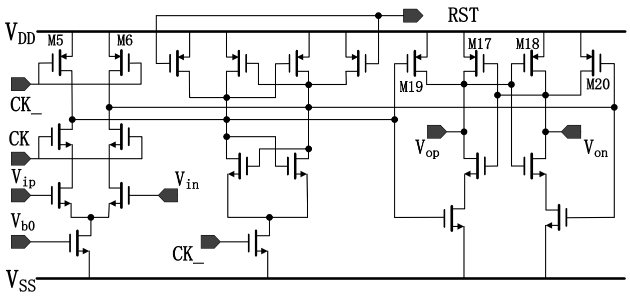


Fig. 9. Schematic of the comparator circuit.

A. Comparator

The comparator is implemented in a 180-nm CMOS process with 1.8-V power supply, shown in Fig. 9. There are 12 design variables for this circuit, representing transistor widths. Both intradie and interdie process variations are considered. Three design specifications are listed as

$$\begin{cases} \text{V}_{\text{off}} \leq 30 \text{ mV} \\ \text{V}_{\text{sen}} \leq 2 \text{ mV} \\ \text{speed} \geq 1 \text{ GHz} \end{cases} \tag{35}$$

where V_{off} denotes the offset voltage with 200-MHz sampling frequency at 30 °C. V_{sen} indicates the ability of comparator to distinguish input signals, and *speed* means the maximum operating frequency.

The proposed approach is compared with [11] and [12] under the same design space. Fig. 10 shows the obtained prior solutions, where $D1$-axis, $D2$-axis, and $D3$-axis are the widths of transistor M17, M19, and M5, respectively, (see Fig. 9). We define the golden solutions as the designs with yields higher than 99%. We can see that some prior solutions are close to the golden solutions, resulting in an efficient warm start.

The quality and speed comparisons of the experimental results are presented in Table I. We take method [11] as the speed benchmark because it is faster than method [12]. All methods can obtain the golden yield. FTBO can gain a 4.86× speedup over [11], and FTBO+ further increases the speedup to 5.73× with the help of prior knowledge. Such experimental results verified the huge advantage of the proposed methods over the existing methods.

More specifically, in terms of the difference in the use of acquisition functions, the proposed method FTBO incorporating ES with wEI gained a 4.86× speedup over [11] which

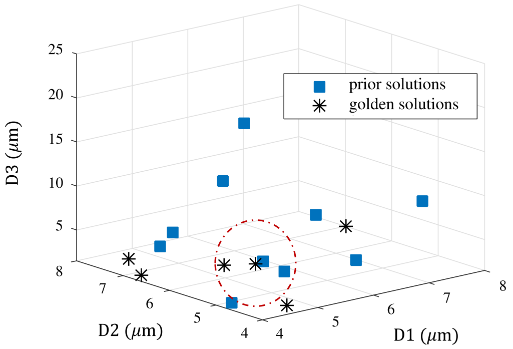


Fig. 10. Distribution of prior solutions for the comparator circuit.

TABLE I
OPTIMIZATION RESULTS AND SPEED COMPARISONS FOR COMPARATOR

Methods		[11]	[12]	FTBO	FTBO+
Yield	Best	99.95%	99.92%	99.98%	99.99%
	Worst	99.19%	99.04%	99.06%	99.18%
	Mean	99.67%	99.42%	**99.74%**	99.67%
	Std. Dev.	0.23%	0.36%	0.30%	0.30%
Number of Simulations	Best	8313	12285	2800	2488
	Worst	94417	103090	17080	14001
	Mean	37408	44807	7695	**6528**
Speedup		1.00	0.83	4.86	**5.73**

TABLE II
TIME COST COMPARISON BETWEEN OPTIMIZATION AND SPICE SIMULATION FOR COMPARATOR

Time of Optimization	freeze-thaw GP training	249s
	wEI optimization	47s
	ES calculation	19s
	Number of iterations	327
	Total optimization time	103005s
Time of SPICE Simulation	Single SPICE simulation	15s
	Number of simulations	37408
	Total simulation time	561120s
Optimization time / SPICE simulation time		18.36%

uses wEI only, and 5.82× speedup over [12] which uses ES only, demonstrating the effectiveness of the combination of wEI and ES in our method.

In each optimization iteration, our method needs 249 s for building the freeze–thaw GP model, 47 s for wEI optimization, and extra 19 s for ES calculation. So our method introduces 6.42% [19 s/(249 s + 47 s)] extra time overhead in optimization as shown in Table II. Therefore, the impact of extra ES calculation on the whole optimization is negligible.

In this case, the number of iterations of yield optimization is 327, so the total optimization time is 103 005 s, i.e., (249 s + 47 s + 19 s) × 327. However, the number of invoked SPICE simulations is 37408, and each SPICE simulation needs 15 s, so the total SPICE simulation time is 561 120 s (15 s × 37 408). Thus, the time of calculating wEI and ES in yield optimization is 18.36% of the SPICE simulation time, which accounts for a relatively small percentage, though the optimization algorithm is currently implemented with the slow Python language. Certainly, SPICE simulation time is highly related to the circuit type, circuit scale and simulation type, etc. Therefore, this ratio of optimization time over SPICE simulation time varies in a wide range.

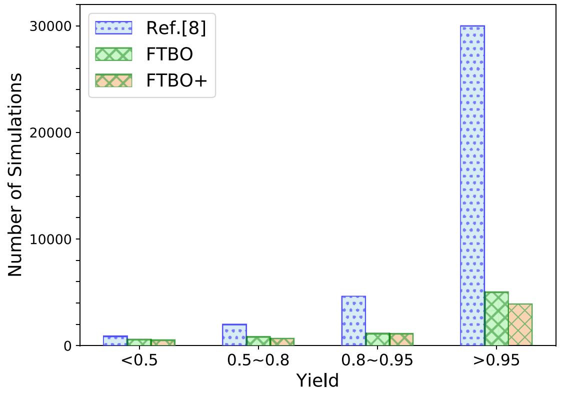


Fig. 11. Comparison of simulation resource allocation between [11] and the two proposed methods in the comparator case.

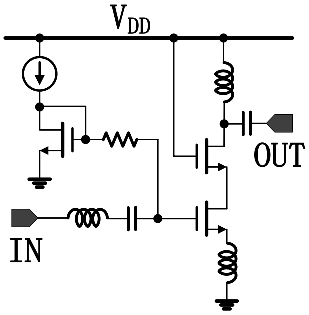


Fig. 12. Schematic of the low noise amplifier circuit.

Fig. 11 further reveals the advantages of the proposed method with the yield bins, where there are four yield bins on the x-axis, and the y-axis is the number of simulations located in the bins. Clearly, [11] can effectively control the analysis accuracy of low-yield designs (yield $\leq$ 95%), i.e., applying coarse yield estimations at these designs with a small number of simulations. However, for high-yield designs, [11] wastes too many simulation resources. FTBO and FTBO+ reduce the simulation number in both low- and high-yield regions, especially in the high-yield region. Concretely, FTBO and FTBO+ reduce the simulation number in the low-yield region by 65% and 68%, respectively. In the high-yield region, FTBO and FTBO+ further cut down the simulation number by 83% and 87%. This result shows that FTBO and FTBO+ are more efficient in simulation resource allocation. Since the simulations allocated to high-yield designs account for more than 80% of the total, the reasonable allocation by freeze–thaw technique in the high-yield region contributes the most to the 4.86× and 5.73× speedup.

B. Low Noise Amplifier

The second case is a low noise amplifier implemented in a 180-nm CMOS process with 3.3-V power supply. Fig. 12 shows the schematic of this radio frequency circuit. 13 design variables are considered in this case, including sizes of transistors, values of capacitors, resistors and inductors. Both intradie and interdie process variations are taken into account. Three specifications, including Gain, noise figure NF, and third-order

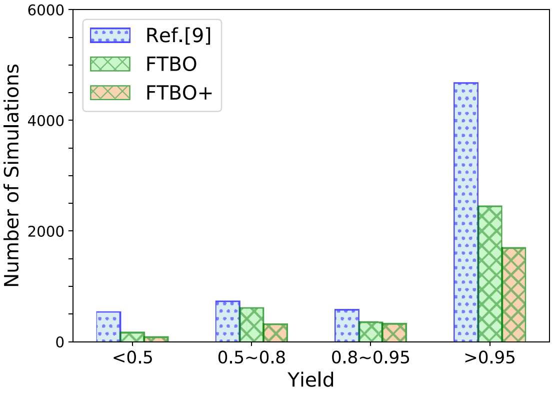


Fig. 13. Comparison of simulation resource allocation between [12] and the two proposed methods in the low noise amplifier case.

TABLE III
OPTIMIZATION RESULTS AND SPEED COMPARISONS FOR LOW NOISE AMPLIFIER

Methods		[11]	[12]	FTBO	FTBO+
Yield	Best	99.89%	99.99%	99.99%	99.99%
	Worst	99.14%	99.73%	99.27%	99.63%
	Mean	99.52%	99.88%	99.80%	99.83%
	Std. Dev.	0.31%	0.09%	0.24%	0.10%
Number of Simulations	Best	3189	3371	1540	1675
	Worst	25369	10906	7189	4707
	Mean	13934	6547	3651	**2646**
Speedup		0.47	1.00	1.79	**2.47**

intercept point IIP3 are listed as

$$\begin{cases} \text{Gain} \geq 20 \text{ dB} \\ \text{NF} \leq 2.3 \text{ dB} \\ \text{IIP3} \geq -10 \text{ dBm.} \end{cases} \tag{36}$$

Table III lists the experimental quality and speed comparisons of optimizations. Method [12] is regarded as the benchmark this time for it is faster than method [11]. Again, all four methods can obtain the golden yield. FTBO and FTBO+ can gain 1.79× and 2.47× speedup over [12], respectively. The comparison of simulation resource allocation between [12] and the two proposed methods is shown in Fig. 13. A similar result can be found that both methods significantly cut down the simulation costs of all yield bins, particularly in the high-yield bin.

C. Three-Stage Amplifier

The third case is a three-stage amplifier implemented in a 0.35-μm CMOS process [43], shown in Fig. 14. There exist 24 design variables corresponding to sizes of transistors, values of capacitors, resistors and biasings. Both intradie and interdie process variations are considered. Four specifications including gain margin GM, gain-bandwidth GBW, phase margin PM and quiescent current I_q at 27 °C are listed as

$$\begin{cases} \text{GM} \geq 20 \text{ dB} \\ \text{GBW} \geq 0.9 \text{ MHz} \\ \text{PM} \geq 50^\circ \\ I_q \leq 70\ \mu\text{A.} \end{cases} \tag{37}$$

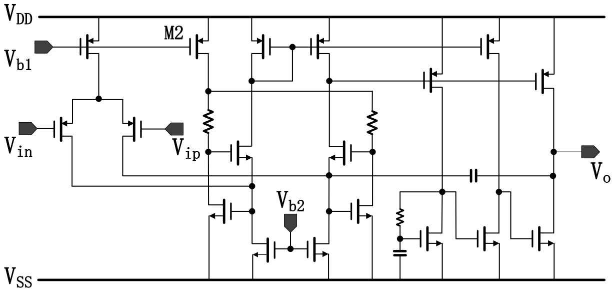


Fig. 14. Schematic of the three-stage amplifier circuit.

TABLE IV
OPTIMIZATION RESULTS AND SPEED COMPARISONS FOR THREE-STAGE AMPLIFIER (0.35 μM)

Methods		[11]	[12]	FTBO	FTBO+
Yield	Best	99.99%	99.99%	99.99%	99.99%
	Worst	99.49%	99.87%	99.42%	99.93%
	Mean	99.81%	99.97%	99.79%	99.96%
	Std. Dev.	0.17%	0.04%	0.17%	0.02%
Number of Simulations	Best	2993	3809	1411	1790
	Worst	20552	12835	2849	2270
	Mean	6086	8350	2083	**1890**
Speedup		1.00	0.73	2.92	**3.22**

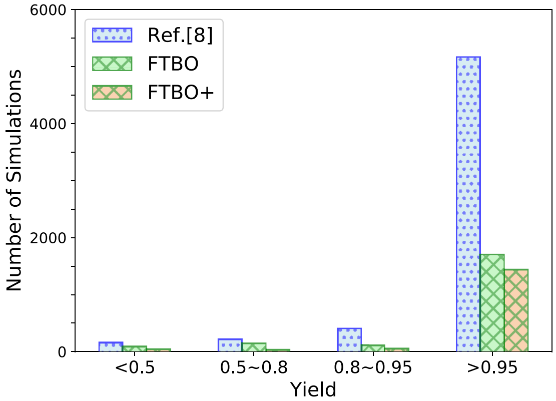


Fig. 15. Comparison of simulation resource allocation between [11] and the two proposed methods in the three-stage amplifier (0.35 μm) case.

Table IV presents the experimental quality and speed comparisons of all four methods. Results of [11] and [12] are taken from [11] and [12], respectively. Method [11] is regarded as the benchmark for it is faster than method [12]. Again, FTBO and FTBO+ can achieve 2.92× and 3.22× speedup over [11]. Fig. 15 shows the comparison of simulation resource allocation between [11] and the two proposed methods.

In order to validate the effectiveness of the proposed method on more advanced technology, this three-stage-amplifier is ported to a 65-nm process with the same design space for yield optimization. The specifications become

$$\begin{cases} \text{GM} \geq 20 \text{ dB} \\ \text{GBW} \geq 0.9 \text{ MHz} \\ \text{PM} \geq 40^\circ \\ I_q \leq 80\ \mu\text{A.} \end{cases} \tag{38}$$

Experimental quality and speed comparisons are shown in Table V. Method [12] is regarded as the benchmark this time

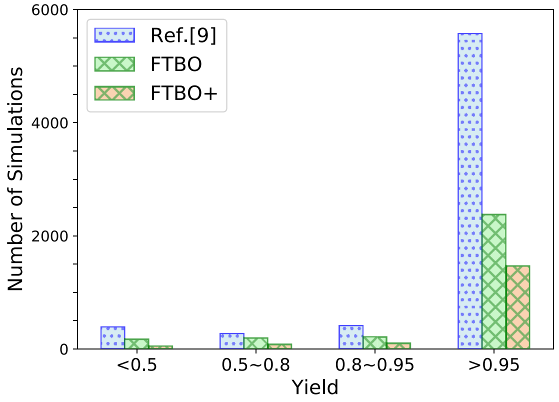


Fig. 16. Comparison of simulation resource allocation between [12] and the two proposed methods in the three-stage amplifier (65 nm) case.

TABLE V
OPTIMIZATION RESULTS AND SPEED COMPARISONS FOR THREE-STAGE AMPLIFIER (65 NM)

Methods		[11]	[12]	FTBO	FTBO+
Yield	Best	99.99%	99.99%	99.99%	99.99%
	Worst	99.08%	99.49%	99.41%	99.89%
	Mean	99.62%	99.87%	99.83%	**99.94%**
	Std. Dev.	0.35%	0.16%	0.18%	0.03%
Number of Simulations	Best	2058	1579	1728	1755
	Worst	21190	22224	7183	3131
	Mean	10228	6736	3006	**1952**
Speedup		0.66	1.00	2.24	**3.45**

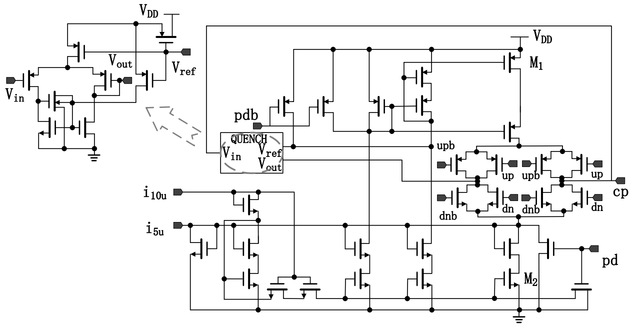


Fig. 17. Schematic of the charge pump circuit.

for it is faster than method [11]. Similarly, FTBO and FTBO+ achieve 2.24× and 3.45× speedup over [12]. The comparison of simulation resource allocation between [12] and the two proposed methods is shown in Fig. 16.

D. Charge Pump

As shown in Fig. 17, the fourth case is a charge pump implemented in a 40-nm CMOS process. 36 design variables are considered in this case. Both intradie and interdie process variations are taken into account. Five specifications, including *diff*1, *diff*2, *diff*3, *diff*4, and *deviation* are listed as

$$\begin{cases} \text{diff1} \leq 25\ \mu\text{A} \\ \text{diff2} \leq 25\ \mu\text{A} \\ \text{diff3} \leq 10\ \mu\text{A} \\ \text{diff4} \leq 10\ \mu\text{A} \\ \text{deviation} \leq 6\ \mu\text{A} \end{cases} \tag{39}$$

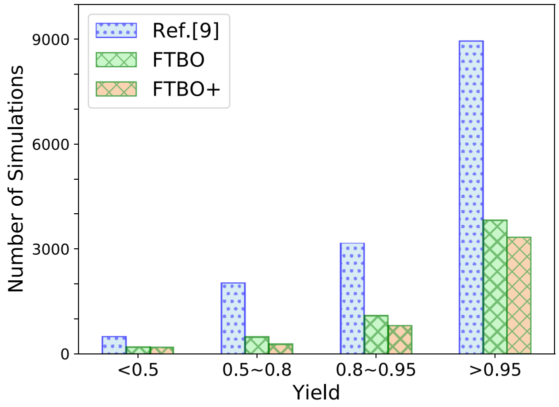


Fig. 18. Comparison of simulation resource allocation between [12] and the two proposed methods in the charge pump case.

TABLE VI
OPTIMIZATION RESULTS AND SPEED COMPARISONS FOR CHARGE PUMP

Methods		[11]	[12]	FTBO	FTBO+
Yield	Best	99.47%	99.94%	99.93%	99.94%
	Worst	99.02%	99.32%	99.06%	99.10%
	Mean	99.16%	99.65%	99.47%	99.56%
	Std. Dev.	0.16%	0.18%	0.32%	0.29%
Number of Simulations	Best	5976	4234	3281	2229
	Worst	60223	22167	10524	12000
	Mean	28163	14581	5514	**4834**
Speedup		0.52	1.00	2.64	**3.02**

where

$$\begin{cases} \text{diff1} = I_{M_1,\max} - I_{M_1,\text{avg}} \\ \text{diff2} = I_{M_1,\text{avg}} - I_{M_1,\min} \\ \text{diff3} = I_{M_2,\max} - I_{M_2,\text{avg}} \\ \text{diff4} = I_{M_2,\text{avg}} - I_{M_2,\min} \\ \text{deviation} = \left|I_{M_1,\text{avg}} - 40\ \mu\text{A}\right| + \left|I_{M_2,\text{avg}} - 40\ \mu\text{A}\right|. \end{cases} \tag{40}$$

Table VI lists the experimental quality and speed comparisons of optimizations. Method [12] is regarded as the benchmark for it is faster than method [11]. All four methods can obtain the golden yield. Without loss of accuracy, FTBO and FTBO+ can gain 2.64× and 3.02× speedup over [12], respectively. As these results are similar to other experiments in Section V, this significant improvement validates the effectiveness of the proposed optimization approach on circuits in advance technology. The comparison of simulation resource allocation between [12] and the two proposed methods is shown in Fig. 18. Again, both methods significantly reduce the simulation costs of all yield bins, especially in the high-yield bin.

E. Three-Stage Amplifier With Large Optimization Space

Usually, in practical design, designers do not know exactly about the locations of the high-yield region. Hence, efficiently exploring a very large design space will be meaningful for designers. In the case of the three-stage amplifier in 0.35-μm process, we roughly expand 10× the optimization ranges for each of 24 design variables, and correspondingly the design space volume is expanded by more than 10^{24} times. The maximum number of iterations for all four methods is set to 1000.

TABLE VII
Optimization Results for Three-Stage Amplifier ($0.35\ \mu$M) in the Large Design Space

Methods		[11]	[12]	FTBO	FTBO+
Yield	Best	99.48%	-	-	99.99%
	Worst	0%	-	-	99.63%
	Mean	41.07%	-	-	**99.83%**
	Std. Dev.	42.87%	-	-	0.10%
Avg. # Sim		17004	-	-	4140
Speedup		1.00	-	-	4.11
# Success		1/10	0/10	0/10	**10/10**

The results in Table VII show that both [12] and FTBO fail to find the optimal design within the ten restarts. Method [11] has one successful result out of ten restarts, and the number of simulations is 17 004. The FTBO+ can find the golden-yield designs at all ten restarts with an average of 4140 simulations. Clearly, the extracted prior knowledge, i.e., exploring the domains with good performance at the TT-corner, will help a lot for yield optimization in large design space.

VI. Conclusion

In this article, a novel and efficient yield optimization approach was proposed for analog circuits. The yield analysis was integrated into the exploration process of Bayesian optimization. The freeze–thaw Bayesian optimization technique was utilized to automatically guide the search in the design space and gradually improve the analysis accuracy in the process space. To further accelerate the yield optimization convergence, a novel performance optimization problem was formulated and solved to mine prior knowledge. Compared with the state-of-the-art methods, the experimental results demonstrated that the proposed method can gain a 2.47×–5.73× speedup without loss of accuracy.

References

[1] G. Gielen, T. Eeckelaert, E. Martens, and T. McConaghy, "Automated synthesis of complex analog circuits," in *Proc. Eur. Conf. Circuit Theory Design*, 2007, pp. 20–23.

[2] T. Chen, Q. Sun, and B. Yu, "Machine learning in nanometer AMS design-for-reliability," in *Proc. ASICON*, 2021, pp. 1–4.

[3] R. Schwencker, F. Schenkel, M. Pronath, and H. Graeb, "Analog circuit sizing using adaptive worst-case parameter sets," in *Proc. DATE*, 2002, pp. 581–585.

[4] M. Barros, J. Guilherme, and N. Horta, "Analog circuits optimization based on evolutionary computation techniques," *Integration*, vol. 43, no. 1, pp. 136–155, 2010.

[5] 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 Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 24, no. 9, pp. 1372–1380, Sep. 2005.

[6] S. P. Mohanty and E. Kougianos, "Incorporating manufacturing process variation awareness in fast design optimization of nanoscale CMOS VCOs," *IEEE Trans. Semicond. Manuf.*, vol. 27, no. 1, pp. 22–31, Feb. 2014.

[7] D. Ghai, S. P. Mohanty, and E. Kougianos, "Design of parasitic and process-variation aware nano-CMOS RF circuits: A VCO case study," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 17, no. 9, pp. 1339–1342, Sep. 2009.

[8] B. Liu, F. V. Fernández, and G. G. E. Gielen, "Efficient and accurate statistical analog yield optimization and variation-aware circuit sizing based on computational intelligence techniques," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 30, no. 6, pp. 793–805, Jun. 2011.

[9] I. Guerra-Gomez, E. Tlelo-Cuautle, and L. G. de la Fraga, "OCBA in the yield optimization of analog integrated circuits by evolutionary algorithms," in *Proc. ISCAS*, 2015, pp. 1933–1936.

[10] M. Wang, F. Yang, C. Yan, X. Zeng, and X. Hu, "Efficient Bayesian yield optimization approach for analog and SRAM circuits," in *Proc. DAC*, 2017, pp. 1–6.

[11] M. Wang *et al.*, "Efficient yield optimization for analog and SRAM circuits via Gaussian process regression and adaptive yield estimation," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 37, no. 10, pp. 1929–1942, Oct. 2018.

[12] S. Zhang, F. Yang, D. Zhou, and X. Zeng, "Bayesian methods for the yield optimization of analog and SRAM circuits," in *Proc. ASP-DAC*, 2020, p. 11.

[13] S. Basu, B. Kommineni, and R. Vemuri, "Variation-aware macromodeling and synthesis of analog circuits using spline center and range method and dynamically reduced design space," in *Proc. VLSI Design*, 2009, pp. 433–438.

[14] V. P. Yanambaka, S. P. Mohanty, E. Kougianos, D. Ghai, and G. Ghai, "Process variation analysis and optimization of a FinFET-based VCO," *IEEE Trans. Semicond. Manuf.*, vol. 30, no. 2, pp. 126–134, May 2017.

[15] O. Okobiah, S. Mohanty, and E. Kougianos, "Fast design optimization through simple kriging metamodeling: A sense amplifier case study," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 22, no. 4, pp. 932–937, Apr. 2014.

[16] O. Garitselov, S. P. Mohanty, and E. Kougianos, "A comparative study of metamodels for fast and accurate simulation of nano-CMOS circuits," *IEEE Trans. Semicond. Manuf.*, vol. 25, no. 1, pp. 26–36, Feb. 2012.

[17] K. Swersky, J. Snoek, and R. P. Adams, "Freeze—thaw Bayesian optimization," 2014, *arXiv:1406.3896*.

[18] T. Domhan, J. T. Springenberg, and F. Hutter, "Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves," in *Proc. IJCAI*, 2015, pp. 3460–3468.

[19] S. Sun, X. Li, H. Liu, K. Luo, and B. Gu, "Fast statistical analysis of rare circuit failure events via scaled-sigma sampling for high-dimensional variation space," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 34, no. 7, pp. 1096–1109, Jul. 2015.

[20] H. E. Graeb, *Analog Design Centering and Sizing*, vol. 64. Berlin, Germany: Springer, 2007.

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

[22] J. Mockus, V. Tiesis, and A. Zilinskas, "The application of Bayesian methods for seeking the extremum," in *Toward Global Optimization*, vol. 2, L. Dixon and G. Szego, Eds. Amsterdam, The Netherlands: Elsevier, 1978.

[23] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian optimization of machine learning algorithms," in *Proc. NIPS*, 2012, pp. 2951–2959.

[24] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. De Freitas, "Taking the human out of the loop: A review of Bayesian optimization," *Proc. IEEE*, vol. 104, no. 1, pp. 148–175, Jan. 2016.

[25] X. Wang *et al.*, "An efficient and robust yield optimization method for high-dimensional SRAM circuits," in *Proc. DAC*, 2020, pp. 1–6.

[26] O. Chapelle and L. Li, "An empirical evaluation of Thompson sampling," in *Proc. NIPS*, 2011, pp. 2249–2257.

[27] J. Dennis and V. Torczon, "Managing approximation models in optimization," in *Multidisciplinary Design Optimization: State-of-the-Art*. Philadelphia, PA, USA: SIAM, 1997.

[28] W. Scott, P. Frazier, and W. Powell, "The correlated knowledge gradient for simulation optimization of continuous parameters using Gaussian process regression," *SIAM J. Optim.*, vol. 21, no. 3, pp. 996–1026, 2011.

[29] A. Budak, M. Gandara, W. Shi, D. Pan, N. Sun, and B. Liu, "An efficient analog circuit sizing method based on machine learning assisted global optimization," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, early access, May 18, 2021, doi: 10.1109/TCAD.2021.3081405.

[30] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *J. Global Optim.*, vol. 13, pp. 455–492, Dec. 1998.

[31] H. Wang *et al.*, "GCN-RL circuit designer: Transferable transistor sizing with graph neural networks and reinforcement learning," in *Proc. DAC*, 2020, pp. 1–6.

[32] W. Lyu *et al.*, "An efficient Bayesian optimization approach for automated optimization of analog circuits," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 65, no. 6, pp. 1954–1967, Jun. 2018.

[33] M. Schonlau, W. J. Welch, and D. R. Jones, "Global versus local search in constrained optimization of computer models," in *Lecture Notes-Monograph Series*. Hayward, CA, USA: Inst. Math. Stat., 1998.

[34] P. Hennig and C. J. Schuler, "Entropy search for information-efficient global optimization," *J. Mach. Learn. Res.*, vol. 13, pp. 1809–1837, Jun. 2012.

[35] A. I. Cowen-Rivers *et al.*, "HEBO: Heteroscedastic evolutionary Bayesian optimisation," 2020, *arXiv:2012.03826v1*.

[36] A. Nieuwoudt and Y. Massoud, "Multi-level approach for integrated spiral inductor optimization," in *Proc. DAC*, 2005, pp. 648–651.

[37] G. Huang, L. Qian, S. Saibua, D. Zhou, and X. Zeng, "An efficient optimization based method to evaluate the DRV of SRAM cells," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 60, no. 6, pp. 1511–1520, Jun. 2013.

[38] 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 *Proc. DATE*, 2016, pp. 1417–1422.

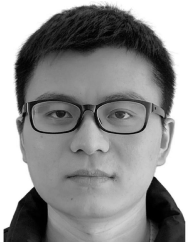
[39] J. Nocedal and S. Wright, *Numerical Optimization*, New York, NY, USA: Springer, 2006.

[40] K. Swersky, J. Snoek, and R. P. Adams, "Multi-task Bayesian optimization," in *Proc. NIPS*, 2013, pp. 2004–2012.

[41] B. Liu, D. Zhao, P. Reynaert, and G. G. E. Gielen, "GASPAD: A general and efficient mm-Wave integrated circuit synthesis method based on surrogate model assisted evolutionary algorithm," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 33, no. 2, pp. 169–182, Feb. 2014.

[42] S. Zhang *et al.*, "An efficient multi-fidelity Bayesian optimization approach for analog circuit synthesis," in *Proc. DAC*, 2019, pp. 1–6.

[43] Z. Yan, P.-I. Mak, M.-K. Law, and R. P. Martins, "A 0.016-mm^2 144-μW three-stage amplifier capable of driving 1-to-15 NF capacitive load with >0.95-MHz GBW," *IEEE J. Solid-State Circuits*, vol. 48, no. 2, pp. 527–540, Feb. 2013.



Xiaodong Wang received the B.S. degree from the Department of Physics, Fudan University, Shanghai, China, in 2017, where he is currently pursuing the Ph.D. degree with the State Key Laboratory of Application Specific Integrated Circuits and System, Microelectronics Department.

His current research interests include yield-related modeling and optimization.

Changhao Yan (Member, IEEE) received the B.E. and M.E. degrees from the Huazhong University of Science and Technology, Wuhan, China, in 1996 and 2002, respectively, and the Ph.D. degree in computer science from Tsinghua University, Beijing, China, in 2006.

He is currently a Professor with the School of Microelectronics, Fudan University, Shanghai, China. His current research interests include analog circuit automation, yield analysis, parasitic parameter extraction, and design for manufacturability.

Yuzhe Ma (Member, IEEE) received the B.E. degree from the Department of Microelectronics, Sun Yat-sen University, Guangzhou, China, in 2016, and the Ph.D. degree from the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, in 2020.

He is currently an Assistant Professor with Microelectronics Thrust, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou. His research interests include agile VLSI design methodologies, machine learning-aided VLSI design, and hardware-friendly machine learning.

Dr. Ma received the Best Paper Awards from ICCAD 2021, ASPDAC 2021, and ICTAI 2019 and the Best Paper Award Nomination from ASPDAC 2019.

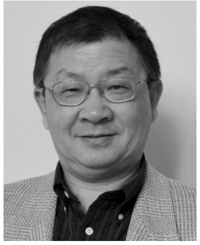
Bei Yu (Member, IEEE) received the Ph.D. degree from the University of Texas at Austin, Austin, TX, USA, in 2014.

He is currently an Associate Professor with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong.

Dr. Yu received the eight Best Paper Awards from ICCAD 2021 and 2013, ASPDAC 2021 and 2012, ICTAI 2019, *Integration the VLSI Journal* in 2018, ISPD 2017, SPIE Advanced Lithography Conference 2016, and six ICCAD/ISPD Contest Awards. He is an Editor of IEEE Technical Committee on Cyber–Physical Systems Newsletter. He has served as the TPC Chair for ACM/IEEE Workshop on Machine Learning for CAD and in many journal editorial boards and conference committees.

Fan Yang (Member, IEEE) received the B.S. degree from Xi'an Jiaotong University, Xi'an, China, in 2003, and the Ph.D. degree from Fudan University, Shanghai, China, in 2008.

He is currently a Professor with the Microelectronics Department, Fudan University. His research interests include model order reduction, circuit simulation, high-level synthesis, yield analysis, and design for manufacturability.



Dian Zhou (Senior Member, IEEE) received the B.S. degree in physics and the M.S. degree in electrical engineering from Fudan University, Shanghai, China, in 1982 and 1985, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Illinois at Urbana–Champaign, Urbana, IL, USA, in 1990.

He joined the University of North Carolina at Charlotte, Charlotte, NC, USA, as an Assistant Professor in 1990, where he became an Associate Professor in 1995. He joined the University of Texas at Dallas, Richardson, TX, USA, as a Full Professor in 1999. His research interests include high-speed VLSI systems, CAD tools, mixed-signal ICs, and algorithms.

Xuan Zeng (Senior Member, IEEE) received the B.S. and Ph.D. degrees in electrical engineering from Fudan University, Shanghai, China, in 1991 and 1997, respectively.

She is currently a Full Professor with the Microelectronics Department, Fudan University, where she served as the Director of the State Key Laboratory of Application Specific Integrated Circuits (ASIC) and Systems from 2008 to 2012. She was a Visiting Professor with the Department of Electrical Engineering, Texas A&M University, College Station, TX, USA, and the Microelectronics Department, Technische Universiteit Delft, Delft, The Netherlands, in 2002 and 2003, respectively. Her current research interests include analog circuit modeling and synthesis, design for manufacturability, high-speed interconnect analysis and optimization, and circuit simulation.

Prof. Zeng received the Changjiang Distinguished Professor with the Ministry of Education Department of China in 2014, the Chinese National Science Funds for Distinguished Young Scientists in 2011, the First-Class of Natural Science Prize of Shanghai in 2012, the 10th For Women in Science Award in China in 2013, and the Shanghai Municipal Natural Science Peony Award in 2014. She received the Best Paper Award from the 8th IEEE Annual Ubiquitous Computing, Electronics and Mobile Communication Conference 2017. She is an Associate Editor of IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—PART II: EXPRESS BRIEFS, IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, and *ACM Transactions on Design Automation of Electronic Systems*.