# An Efficient High-Frequency Linear RF Amplifier Synthesis Method Based on Evolutionary Computation and Machine Learning Techniques

Bo Liu, Noël Deferm, Dixian Zhao, Patrick Reynaert, and Georges G. E. Gielen, *Fellow, IEEE*

***Abstract*—Existing radio frequency (RF) integrated circuit (IC) design automation methods focus on the synthesis of circuits at a few GHz, typically less than 10 GHz. That framework is difficult to apply to RF IC synthesis at mm-wave frequencies (e.g., 60–100 GHz). In this paper, a new method, called efficient machine learning-based differential evolution, is presented for mm-wave frequency linear RF amplifier synthesis. By using electromagnetic (EM) simulations to evaluate the key passive components, the evaluation of circuit performances is accurate and solves the limitations of parasitic-included equivalent circuit models and predefined layout templates used in the existing synthesis framework. A decomposition method separates the design variables that require expensive EM simulations and the variables that only need cheap circuit simulations. Hence, a low-dimensional expensive optimization problem is generated. By the newly proposed core algorithm integrating adaptive population generation, naive Bayes classification, Gaussian process and differential evolution, the generated low-dimensional expensive optimization problem can be solved efficiently (by the online surrogate model), and global search (by evolutionary computation) can be achieved. A 100 GHz three-stage differential amplifier is synthesized in a 90 nm CMOS technology. The power gain reaches 10 dB with more than 20 GHz bandwidth. The synthesis costs only 25 h, having a comparable result and a nine times speed enhancement compared with directly using the EM simulator and global optimization algorithms.**

***Index Terms*—Differential evolution, efficient global optimization, expensive black-box optimization, Gaussian process, mm-wave frequency, radio frequency (RF) circuit synthesis.**

## I. Introduction

In recent years, the demand for high-data-rate wireless communication systems is constantly increasing. However, low-GHz radio frequency (RF) integrated circuits (ICs) (i.e., below 5 GHz) are not able to support these high-data-rate communications [1]. Hence, the design and optimization methods for mm-wave RF ICs are attracting a lot of attention recently. In particular, research on RF building blocks from 40 GHz to 120 GHz and beyond is increasing drastically and moving to industrial applications. For example, 60 GHz RF ICs are widely used for uncompressed HDTV and 94 GHz RF IC is used for microwave imaging systems. On the other hand, the challenge is that parasitic-aware lumped equivalent circuit models for passive components (e.g., inductor, transformer) that accurately match the electromagnetic (EM) simulation results, are often difficult to find at these frequencies [2]. Some designers rely on experience and simulation verification when designing high-frequency RF ICs. However, due to the high performance and tightening time-to-market requirements, the "experience and trial" method is often not good enough. Powerful and efficient optimization techniques are therefore largely needed.

Existing RF IC design automation methods focus on low-GHz synthesis [3]–[11]. The framework of most of these methods is shown in Fig. 1. Compared with the low-frequency analog circuit sizing flow, a key part is the generation of the parasitic-aware model of passive components. In RF IC design at low-GHz frequencies, a simple lumped model is often extracted to mimic the behavior of the key passive components (transformer, inductor). Regression methods are then used to fit the (calibrated) EM simulation results (S-parameters) to parasitic-included equivalent circuit models. The generated passive component models are accurate at low-GHz frequencies and computationally efficient. To make the parasitic-aware model reliable to provide the correct performances for different design parameters, a strictly enforced layout template is often necessary. References [7] and [8] used the parasitic corner, rather than a strict layout template, to improve the flexibility of the generated layout for circuits below 10 GHz, yielding good results. In the development of the optimization kernel, evolutionary algorithms (EAs) are introduced in RF IC synthesis to achieve global search, getting very good results. Reference [11] used particle swarm optimization and [10] introduced the nondominated genetic algorithm to RF IC synthesis in order to achieve multiobjective optimization.

Although these works make great contributions, they focus on low-GHz RF synthesis, and the framework in Fig. 1 is difficult to extend to mm-wave frequency synthesis. The reason is that the quality of the parasitic-aware models for key passive components is determined by both the regression method and the equivalent-circuit model. The inductor and transformer equivalent circuit models in the existing methods

Manuscript received August 5, 2011; revised October 21, 2011 and December 16, 2011; accepted January 24, 2012. Date of current version June 20, 2012. This work was supported by a bilateral agreement scholarship of Katholieke Universiteit Leuven, Leuven, Belgium, and Tsinghua University, Beijing, China. This paper was recommended by Associate Editor H. E. Graeb.

The authors are with Katholieke Universiteit Leuven, Leuven 3001, Belgium (e-mail: bo.liu@esat.kuleuven.be; liu_bo765@yahoo.com.cn; noel.deferm@esat.kuleuven.be; dixian.zhao@esat.kuleuven.be; patrick.reynaert@esat.kuleuven.be; georges.gielen@esat.kuleuven.be).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCAD.2012.2187207

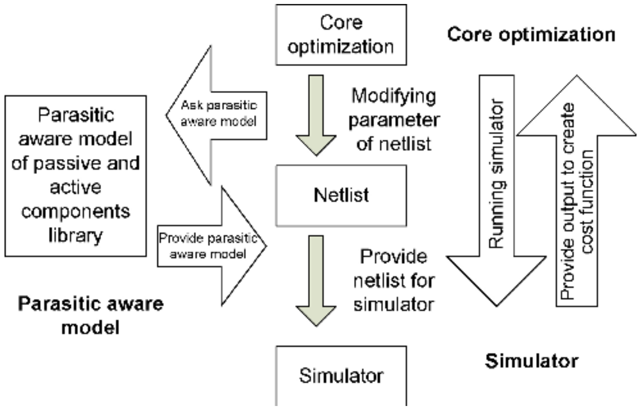
Core optimization
Core optimization
Parasitic aware model of passive and active components library
Ask parasitic aware model
Modifying parameter of netlist
Netlist
Running simulator
Provide output to create cost function
Provide parasitic aware model
Provide netlist for simulator
Parasitic aware model
Simulator
Simulator

Fig. 1. Framework of parasitic-aware optimization to RF IC (from [4]).

are simple and work well in low-GHz cases after good regression. At frequencies above 60 GHz, due to the distributed effects at these mm-wave frequencies, it is not accurate to present a passive component by a lumped model over a wide bandwidth. Besides, it is usually difficult and time consuming to fit the S-parameters to a distributed model, which includes quite a number of resistors, inductors, capacitors, and coupling factors. Hence, even assuming that the regression method works well, the parasitic-aware passive component model is still difficult to generate, due to the lack of good lumped models at mm-wave frequencies.

To address this problem, manual designers often directly include S-parameters into the circuit simulations. In this paper, we also directly include S-parameters by incorporating EM simulation in the optimization loop, which has the advantage of achieving very good accuracy and generality. The EA is used to achieve global search. Combining these two techniques, high-quality solutions can be obtained. However, the challenge is that the EM simulations included in each performance evaluation are very CPU time expensive and that the standard EA needs more function evaluations compared with nonpopulation-based optimization algorithms, although the optimization ability is much higher [12]. Hence, computational efficiency becomes the main problem. In this paper, we address the efficiency problem by proposing a different approach that exploits techniques from computational intelligence: a hierarchical machine learning-based evolutionary optimization mechanism is designed. Compared with directly using a EA with EM simulations in the loop, comparable results can be obtained, but the new approach is nine times faster in the tested 100 GHz three-stage differential amplifier.

Based on the above ideas, we then propose a new synthesis framework for mm-wave RF linear amplifiers, called efficient machine learning-based differential evolution (EMLDE). This synthesis framework focuses on linear amplifiers, e.g., wideband amplifiers. Nonlinear analysis is beyond the scope of this paper. The nonlinear simulations at mm-wave frequencies sometimes cannot converge, and additional methods are needed to address the problem. This paper aims to:

1) develop the first synthesis method for linear mm-wave RF amplifiers beyond 60 GHz starting from a given circuit topology, specifications and some hints on layout (e.g., the metal layer to be used, the transistor layout template with different number of fingers);
2) provide highly optimized results comparable to directly using a EA with EM simulations in the optimization loop, which is the best known method on the solution quality aspect;
3) use much less computational effort compared with using the standard EA, and as such make the computational time of the synthesis practical;
4) be general enough for any technology and any frequency in the mm-wave frequency range.

The remainder of this paper is organized as follows. Section II briefly introduces the authors' previous work of the synthesis of passive components for high-frequency RF ICs, some of which ideas are used in this paper. Section III introduces the general ideas and framework of the EMLDE method. Section IV presents the detailed description of the core algorithms in the EMLDE framework. Section V tests the synthesis method on a 100 GHz three-stage amplifier in a 90 nm CMOS technology. The comparison is performed to the method of directly using the EA with EM simulations in the optimization loop, which is the method that can provide the best result with respect to solution quality. A cheap and easy-to-implement comparison and test framework for RF design automation researchers is also constructed based on mathematical benchmark problems in the evolutionary computation (EC) field. Concluding remarks are presented in Section VI.

## II. Memetic Machine Learning-Based Differential Evolution (MMLDE) Method and Basic Techniques

The authors proposed the MMLDE algorithm, for the optimized synthesis of integrated passive components (three to five design variables) in mm-wave frequencies [2]. MMLDE can provide comparable results with the best framework in terms of solution quality, but at far lower computational cost. Some of the key ideas of MMLDE are used in EMLDE, and are briefly described in the following.

### *A. Online Surrogate Model-Based Optimization for Low Dimensional Problems*

Although a EA can achieve global search, it often needs a number of function evaluations. When the function evaluation is expensive, such as when the user needs EM simulation, the efficiency of the synthesis is low. The key idea of MMLDE is to use an online surrogate model to increase the efficiency. An initial Gaussian process (GP)-based surrogate model is constructed first by using $11 \times d - 1$ Latin hypercube (LHS) [13] samples that uniformly cover the design space, where $d$ is the number of design variables. This model can provide a very rough estimation of the performances of the passive component. In optimization, the constructed surrogate model and the prescreening method evaluate the potential of the candidate designs. The candidate designs are ranked and the one with the best potential is selected to perform the EM simulation. The new point will be used to update the surrogate model. We iteratively repeat this process until the termination condition is met. An important advantage of MMLDE is the use of the online surrogate model, which is constructed based on the available data in the optimization process. The promising solutions are selected meanwhile and guide further candidate solution generation. The advantages on efficiency and reliability of online surrogate model-based optimization compared with offline surrogate model-based optimization [14] are described and compared in [2].

On the other hand, the challenge of online surrogate-model-based optimization is that the quality of the surrogate model is not always good, as it is improving gradually. If the training data is little in some area of the design space, especially in the beginning stage, not enough information can be provided to the learning machine, so the surrogate model might not be good enough. When directly using the performance values predicted by the surrogate model to judge the potentials of the candidate designs, the search may go to wrong directions and can finally be trapped in a local optimal point [2], [15]. To solve this problem, the expected improvement (EI) prescreening with a GP-based surrogate model focusing on global search and an artificial neural network-based surrogate model [16] focusing on local search is proposed. This method achieves a good ranking and a high probability of correct selection for promising candidates, even when a good enough surrogate model is not available. More details are in [2].

### *B. GP-Based Machine Learning*

Because a GP-based surrogate model is also used in EMLDE, a brief introduction is provided, and more details are in [2]. GP machine learning [17] assumes that the objective function is a sample of a Gaussian stochastic process. The distribution of the function value of a new point can be predicted by the available points. Both a predicted value and a prediction uncertainty for a new point are provided, which are then used by the prescreening method to select promising points to perform expensive function evaluations. The GP-based surrogate model has a wide application in expensive black-box optimization and gets very good results [15], [18]. We will now explain this.

Suppose that there are $n$ training data $x = (x_1, x_2, \cdots, x_n)$ and their corresponding function values are $y = (y_1, y_2, \cdots, y_n)$. Using the GP model with the correlation function $Corr(x_i, x_j)$, the function value $y(x^*)$ at a new point $x^*$ can be predicted as

$$\hat{y}(x^*) = \hat{\mu} + r^T R^{-1}(y - I\hat{\mu}) \tag{1}$$

where

$$R_{i,j} = Corr(x_i, x_j) \quad i, j = 1, 2, \cdots, n \tag{2}$$

$$r = [Corr(x^*, x_1), Corr(x^*, x_2), \cdots, Corr(x^*, x_n)]^T \tag{3}$$

$$\hat{\mu} = (I^T R^{-1} I)^{-1} I^T R^{-1} y. \tag{4}$$

The mean square error (MSE) of the prediction is

$$MSE(x^*) = \hat{\sigma}^2[I - r^T R^{-1} r + (I - r^T R^{-1} r)^2 (I^T R^{-1} I)^{-1}] \tag{5}$$

where

$$\hat{\sigma}^2 = (y - I\hat{\mu})^T R^{-1}(y - I\hat{\mu})n^{-1}. \tag{6}$$

The potential (using EI prescreening [15]) of the point $x^*$ is calculated as

$$\begin{aligned} E[I(x)] = {} & (f_{\min} - y(x))\Phi\left(\frac{f_{\min} - y(x)}{\sqrt{MSE(x)}}\right) \\ & + \sqrt{MSE(x)}\phi\left(\frac{f_{\min} - y(x)}{\sqrt{MSE(x)}}\right) \end{aligned} \tag{7}$$

where $f_{\min}$ is the current best function value in the available data. $\phi(\cdot)$ is the standard normal density function, and $\Phi(\cdot)$ is the standard normal distribution function. $I(x)$ is the improvement of $f$. EI is essentially the part of the curve of the standard error in the model that lies below the best function value sampled so far. It can be seen that the EI prescreening considers both the predicted value and the possible prediction error. Therefore, the quality of a new candidate is evaluated in a global picture.

### *C. Differential Evolution Algorithm*

The optimization kernel in MMLDE is the differential evolution (DE) algorithm [19], which is also used in EMLDE. The DE algorithm outperforms many other EAs in terms of solution quality and convergence speed [19]. DE uses a simple differential operator to create new candidate solutions and a one-to-one competition scheme to greedily select new candidates.

The $i$th candidate solution in the $d$-dimensional search space at generation $t$ can be represented as

$$x_i(t) = [x_{i,1}, x_{i,2}, \cdots, x_{i,d}]. \tag{8}$$

At each generation $t$, the *mutation* and *crossover* operators are applied to the candidate solutions, and a new population arises. Then, *selection* takes place, and the corresponding candidate solutions from both populations compete to comprise the next generation. The operators are as follows:

$$\text{Mutation: } V_i(t+1) = x_{\text{best}}(t) + \hat{F}_i(x_{r1}(t) - x_{r2}(t)) \tag{9}$$

where indices $r_1$ and $r_2$ ($r_1, r_2 \in \{1, 2, \ldots, NP\}$) are randomly chosen and mutually different, and also different from the current index $i$. $NP$ is the population size. $x_{\text{best}}(t)$ is the best individual of the current population. $\hat{F}$ is the scaling factor. We use a vector composed of Gaussian-distributed random variables with mean value $\mu$ and variance $\sigma$: $\hat{F}_{i,j} = norm(\mu, \sigma)$, $i = 1, 2, \ldots, NP$, $j = 1, 2, \ldots, d$.

Crossover: a trial vector is generated as follows:

$$U_i(t+1) = [u_{i,1}(t+1), \ldots, u_{i,d}(t+1)] \tag{10}$$

$$u_{i,j}(t{+}1) = \begin{cases} v_{i,j}(t+1), & \text{if } (rand(i,j) \le CR) \text{ or } j = randn(i) \\ x_{i,j}(t), & \text{otherwise} \end{cases} \tag{11}$$

where $rand(i, j)$ is an independent random number uniformly distributed in the range [0, 1]. Parameter $randn(i)$ is a randomly chosen index from the set $\{1, 2, \ldots, d\}$. Parameter

$CR \in [0, 1]$ is a user-defined constant called the crossover parameter.

Selection:

$$x_i(t+1) = \begin{cases} U_i(t+1), & \text{if } f(U_i(t+1)) < f(x_i(t)) \\ x_i(t), & \text{otherwise} \end{cases} \tag{12}$$

where the function $f$ is the objective function, i.e., the function to be minimized or maximized. The candidate solution, $x_i(t+1)$, becomes the candidate solution of the new population. Then, the next iteration begins. For more details about the DE algorithm, please see [2] and [19].

## III. The EMLDE Method

Although MMLDE can solve low-dimensional expensive optimization problems very well, the synthesis of mm-wave RF linear amplifiers brings new challenges, which make the MMLDE algorithm not workable in this problem. This is also the "curse of dimensionality" in surrogate model assisted evolutionary algorithms. For the problem of RF amplifier synthesis, one stage of the amplifier often has 10–20 design variables (Section V provides an example). However, most GP-surrogate model assisted evolutionary algorithms normally can handle expensive optimization problems with about five variables very efficiently. Many works of expensive black-box optimization in the computational intelligence field focus on small-scale problems (e.g., [15], [20], [21]). When the number of the dimensions increases, two challenges appear.

1) Solution quality: an initial surrogate model that can roughly approximate the performance of the circuit is often difficult to be constructed with a reasonably small number of initial samplings. Because the initial information is very little, promising areas are hard to be selected correctly even with good prescreening methods.
2) Efficiency: a linear increase of the number of design variables causes an exponential increase of the search space, which requires more training data in the online optimization process. This also lowers the speed of the synthesis considerably, because more samples and iterations are needed and each of them is expensive. In addition, the computational effort to construct the GP model itself increases drastically with the number of design variables and the number of training data.

To address this problem, the new contributions in this paper focus on dimension reduction to transform the original problem to a lower dimensional problem and on the effective solution of this reformulated low-dimensional but more complex problem. The dimension reduction method is introduced in Section III-C and the proposed new algorithm to solve "hard to predict" low-dimensional expensive optimization problems is introduced in Section IV.

The general framework of the EMLDE method for mm-wave RF linear amplifier synthesis is shown in Fig. 2. In the following, the key ideas and main blocks of the flow will be described. The core algorithms will be illustrated in a separate section.

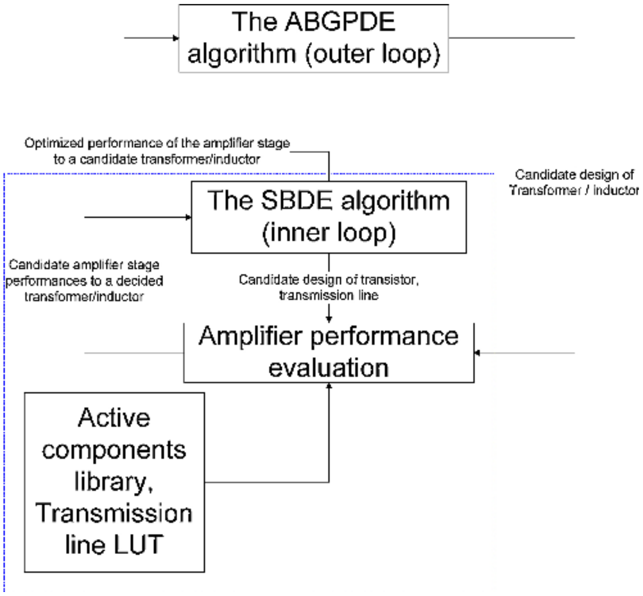
The ABGPDE algorithm (outer loop)
Optimized performance of the amplifier stage to a candidate transformer/inductor
Candidate design of Transformer / Inductor
The SBDE algorithm (inner loop)
Candidate amplifier stage performances to a decided transformer/inductor
Candidate design of transistor, transmission line
Amplifier performance evaluation
Active components library, Transmission line LUT

Fig. 2. Framework of the EMLDE method.

### A. Active Components Library and the Look-Up Table for Transmission Lines

In EMLDE, the active components library is the same as the parasitic-aware active components library in existing low-frequency RF IC synthesis methods, which is shown in Fig. 1. For transmission lines, we use a look-up table to get the S-parameters. In the case of a linear RF amplifier optimizing performances according to S-parameters (e.g., power gain), the computationally expensive parts are: the parasitic extraction of active devices (transistors), the EM simulation of the long transmission lines, and the EM simulation of transformers and inductors. When the performance optimization is based on linear analysis (for many CMOS technologies, nonlinear analysis results, such as output power, are difficult to be optimized due to the high working frequencies), usually the most critical problem is the impedance matching, and the transistors often have clear design rules. For example, a typical method is to use the minimum transistor length and a fixed width, while only the number of fingers is changed. In addition, the transistor layout is decided before any other components in many high-frequency amplifier designs. Hence, we suggest first extracting the parasitics of the transistors with different number of fingers but with fixed width and length beforehand and then directly use the extracted models in full-fletched optimization. Although optimizing the transistors (by changing the number of fingers) with already extracted models is not used in the experiments of this paper because of not necessary, this method is recommended when transistor sizing is necessary in synthesis. The extraction consumes some computational effort, but it is a one-time investment for each technology. Note that changing the number of fingers brings a discrete design variable in the optimization. A quantization technique [19] can be used to make the floating-point-based DE method also workable for mixed continuous and discrete optimization problems.

For the transmission lines, their S-parameters are highly linear. We first sweep the transmission lines with different line widths, lengths and distances between two lines if using differential transmission lines, and then build a look-up table (LUT). Through experiments, we found that the S-parameters generated by interpolation from the LUT have little difference compared with the EM simulation results. The very small difference can only add less than 1% error on the circuit performances in our experiments. Like the active components, the data generation of the transmission lines is also a one-time investment. This solves the efficiency problem of performing online EM simulations to newly generated long transmission lines. Therefore, the most expensive and difficult part remaining is the EM simulation of transformers and inductors, which cannot be solved by the existing methods when going to mm-wave frequencies. This is also the focus of the efficient global optimization algorithm proposed in this paper.

### B. Handling Cascaded Amplifiers

At high frequencies, to obtain a higher gain, the amplifier often includes multiple stages, which are cascaded together. For example, [22] designed the first fully differential 100 GHz CMOS amplifier, which uses six cascaded stages to obtain about 10 dB power gain. In manual design, the designer often copies the design of one stage to construct the cascaded amplifier. This result is less optimal because the impedance matching of each stage is different. In contrast, the synthesis method proposed optimizes the cascaded amplifier stage by stage according to each stage's own impedance matching. In EMLDE, instead of dividing the circuit by active components like in manual design, transformers and inductors are the main objects for circuit division. Even for single-stage amplifiers but with a complex structure, dividing the circuit by key passive components is also workable in most cases. The division rules are: 1) one stage includes one and only one computationally expensive passive component; 2) the components in each stage must be connected together; and 3) there should not exist components that do not belong to any stage, such as the input/output pads. More details of division will be shown by the example in Section V.

### C. Two Optimization Loops

In this subsection, we introduce our method to reduce the number of dimensions to a stage of the RF amplifier.

Usually, the design parameters of a stage of the RF amplifier include the parameters of the transformers or inductors, the parameters of the transistors, and the parameters of the connecting transmission lines. The overall circuit performance is decided by all of them. But with the help of the active components library and the look-up table for transmission lines in Section III-A, only the parameters of the transformers or inductors need expensive EM simulation. In addition, the number of parameters of a transformer or inductor is not large (often four to five). Hence, a natural idea is to separate these design variables. Our method to reformulate the problem is as follows and also shown in Fig. 2.

The parameters of the transformer or inductor are set as the design variables (input), and the performances of the amplifier with the *decided* transformer or inductor and the *corresponding optimized* transistors and transmission lines are the output variables. In this way, the GP-based machine learning can be used for the outer optimization loop to decrease the number of expensive EM simulations. In other words, the original plain optimization problem is reformulated as a hierarchical optimization problem. The outer loop is the optimization of the transformer or inductor parameters, whose function values are the optimized performances of the amplifier stage, which is obtained by the inner optimization loop. The inner loop is the optimization of the transistors and transmission lines for the decided transformer or inductor provided by the outer loop. Although the inner loop needs more computational effort (an optimization is needed, rather than a single simulation), with the efficient models for transistors and transmission lines, and fast S-parameter circuit simulation, the evaluation of the inner function is very cheap. In addition, because of the independence of the candidates in the population of EAs, parallel computation is used to further decrease the computational time. An 8-core CPU is used in this paper. We use the selection-based differential evolution (SBDE) algorithm [23] for the inner optimization. Details are described in Section IV.

However, the price to pay for lowering the number of dimensions of the problem by decomposition is that the GP prediction and the EI prescreening of the potential of a candidate design become more difficult. The reason is that the original performance is explicitly correlated to 10–20 variables, while in the new problem formulation it is predicted by four to five variables only and more than ten variables are hidden. Hence, the problem to be predicted is more complex. Through experiments, we found that only using the standard GP method and EI prescreening is not good enough to make the selection of the promising solutions effectively. This means that more iterations are necessary, which naturally leads to more EM simulations and more inner optimizations. Hence, we propose a new GP surrogate model assisted evolutionary algorithm: adaptive population generation, naive Bayes classification, and Gaussian process-based differential evolution (ABGPDE). Using the ABGPDE algorithm, highly optimized designs can be obtained efficiently, with a solution quality comparable to directly using a global optimization algorithm with EM simulations in the function evaluations. In the test example of a 100 GHz differential amplifier, the speed is nine times faster and the total synthesis time only costs 25 h, which makes the computational time practical. ABGPDE will be described in Section IV and the comparison with the standard GP-model-based efficient global optimization method [15] is shown in Section V.

## IV. Key Algorithms in EMLDE

EMLDE includes decomposition into two optimization loops (see Section III-C): the SBDE algorithm for inner optimization and the ABGPDE algorithm for outer optimization. This section will introduce the ABGPDE and SBDE algorithms. The naive Bayes classifier is a component of ABGPDE, so it is introduced first.

### A. Naive Bayes Classification

As said in Section III-C, by lowering the number of dimensions of the original optimization problem, the drawback is that the performances of the generated low-dimensional expensive optimization problem are more difficult to predict. Naive Bayes classification [24] is used to help the EI prescreening to select the most promising candidate design.

The naive Bayes classification is very efficient and outperforms many existing classification methods, even some newly developed methods [25]. A classifier is a machine that maps the input feature space $F$ to the output class label space $C$. Naive Bayes classification is a supervised learning method, which learns from a training data set of input vectors (input features) and their corresponding classes. In the following, we introduce how the naive Bayes classifier works.

Suppose that the input vector is $d$-dimensional, so we have feature variables from $F_1$ to $F_d$. Each input vector is classified to a class $C_i$ $(i = 1, ..., n)$. For a new input vector $x$, the class it belongs to is decided by the maximum probability of the hypothesis that $x$ belongs to $C_i$, that is

$$class(x) = \arg\max_{C_i} p(C = C_i | F_1 = x_1, \ldots, F_d = x_d). \quad (13)$$

The naive Bayes classifier assumes that each feature $F_k$ $(k = 1, \ldots, d)$ is conditionally independent of every other feature. Hence, the conditional probability $p(C|F_1, \ldots, F_d)$ can be highly simplified to

$$\begin{aligned} p(C|F_1, \ldots, F_d) &= \frac{p(C, F_1, \ldots, F_d)}{p(F_1, \ldots, F_d)} \\ &= \frac{p(C) \prod_{k=1}^{d} p(F_k|C)}{p(F_1, \ldots, F_d)} \end{aligned} \quad (14)$$

where $p(F_1, \ldots, F_d)$ is common to all and does not affect the ranking of (13). In this paper, we assume that the input vector values associated with each class are Gaussian distributed. According to the training data, the mean and variance of the data associated with each class can be calculated. Using the probability density function of the Gaussian distribution and plugging (14) into (13), the corresponding class for a new vector $x$ can be calculated.

Although the basic assumption of independence of all the features is often not accurate enough, the naive Bayes classifier uses the maximum of posteriori rule [24]. Therefore, the classification is decided by the ranking, rather than the accurate estimation of $p(C = C_i|F_1 = x_1, \ldots, F_d = x_d)$. This is the main reason why the naive Bayes classifier can still be very effective, even when using such a simplified assumption.

### B. ABGPDE Algorithm

The proposed ABGPDE algorithm is a surrogate model assisted evolutionary algorithm for low-dimensional expensive optimization problems, especially suitable for problems with difficult to predict data sets. In EMLDE, the ABGPDE algorithm solves the outer loop optimization. The input are the design parameters of the transformer or inductor (four to five dimensions), and the output is the performance of a stage of the linear amplifier with its corresponding optimized transistors and transmission lines. The function evaluation includes EM simulation of the transformer or inductor and the inner optimization loop.

1) *Structure of ABGPDE:* Due to the dimension reduction, the basic structure of MMLDE (see Section II-A) can therefore be used in ABGPDE. The constraint handling method is the static penalty function method [26]. The differences with MMLDE are as follows.

1) The inner optimization is included. Both in initialization and optimization, an inner optimization is performed to each passive component design to obtain the corresponding optimal performance of the amplifier stage.
2) Although the EI prescreening [15] (7) is still used, the evaluation of the potential of the candidates is different from [2], which is described in the next subsection.
3) The population setting is different. In ABGPDE, there are two populations: one is the population containing all the simulated candidates, which is the same as [2]; the other is adaptively constructed in each iteration, which is described in the next subsection.

It can also be mentioned that the EM simulations for the initial points can be done beforehand for one technology, which is also a one-time investment and can be used in all the stages. Only the inner optimization loop for the initial points needs to be done for each stage, because the best matching can be different from stage to stage.

2) *Handling Difficult to Predict Data Set:* After lowering the original plain optimization problem with 10–20 design variables to a hierarchical optimization problem with four to five variables, the predictions are more difficult. Through experiments, we found that using MMLDE [2], a satisfactory solution often needs many iterations, and each iteration is expensive (see Section V-B). The reason is that because of the complexity of the prediction problem, the number of wrong selections using the EI prescreening increases a lot.

We address the problem in two ways: 1) improving the potential evaluation of the candidates; and 2) revising the EA.

For the EI prescreening, at the same time of achieving global search, it also bears the unavoidable risk of not selecting good candidates. For a candidate whose tail of the probability density function is smaller than $f_{\min}$, the possibilities exist both of the candidate being truly a promising one or being not promising but having a large estimation variance. But EI cannot classify them. Hence, rather than improving EI, we add the naive Bayes classification to help EI for this classification. As said above, if the function is continuous, the function values of two points $x_i$ and $x_j$ should be close if they are highly correlated. We use the naive Bayes classifier which only considers the input space $x$ to make classifications. This is a good supplement to GP machine learning which considers both the input space and the output space. If a candidate has a high EI value but is classified into the unpromising points class, there is a high probability that the point has an unpromising function value but with a large estimation variance.

In ABGPDE, we use the mean performance of the current circuit being synthesized (e.g., power gain of two stages of the circuit) for all the candidate designs of the current population as the threshold. The candidates with function value better than the threshold are classified as promising points; otherwise, they are classified as unpromising points. These construct the training data. For a new population, we select the candidate solution with the highest EI value in the promising point class as the most promising one and evaluate it.

Although the naive Bayes classifier helps EI to evaluate the potential of the candidate solutions, it only contributes to the identification of promising solutions. On the other hand, the problem of how to make promising solutions being generated more efficiently is still not answered. The EA we use for expensive optimization is different from the standard EAs. In standard EAs, the solution quality of the population is improving in the evolution process, so beneficial information to generate promising candidates keeps increasing in the consecutive populations. After some iterations, a high percentage of the information in the current population is beneficial to generate a candidate with good quality. In contrast, for surrogate model-assisted optimization, besides the initial samples, only one or few good new individuals are evaluated and added to the population in each iteration to increase the efficiency. Hence, in many occasions, the majority of the population are the initial samples. The goal of the initial samples is to cover the design space, but many of them may not be good solutions. Consequently, the percentage of beneficial information in the consecutive populations increases slowly. In evolutionary optimization, the new population is generated according to the information of the previous population by evolution operators. If the beneficial information in the previous population is less, generating promising candidates is more difficult.

Our idea to solve this problem is to artificially increase the amount of beneficial information by constructing a new population for evolution. In each iteration, we rank all the candidates in the original population and select the top 75% candidates to enter the new population for evolution. The remaining 25% of the new population is filled by randomly selecting the candidates from the top 75% candidates of the original population. This operation may sacrifice the diversity a little bit but causes fast convergence to a satisfactory result, which fits the needs for expensive optimization problems. After all, in EMLDE, because all the simulated candidates are included in the original population, the diversity is quite high. Hence, if we replace some low-quality candidates, the diversity does not decrease too much.

Based on the above ideas, the ABGPDE algorithm is constructed. Experimental results show that ABGPDE enhances the speed and solution quality considerably compared to only using the GP-based surrogate model and EI prescreening, which is the idea of standard efficient global optimization (EGO) in computational intelligence [15]. The experimental comparisons are given in Section V. The flow diagram and the ABGPDE algorithm description will be shown in Section IV-D.

### C. SBDE Algorithm

The SBDE algorithm is used for the inner optimization loop of the synthesis system. The inputs are transistor parameters, transmission line parameters, and DC voltages. The used transformer or inductor and its EM simulation result are provided from the outer loop. The output is the performance of one stage of the RF amplifier with optimized transistors, transmission lines, and biasing voltages.

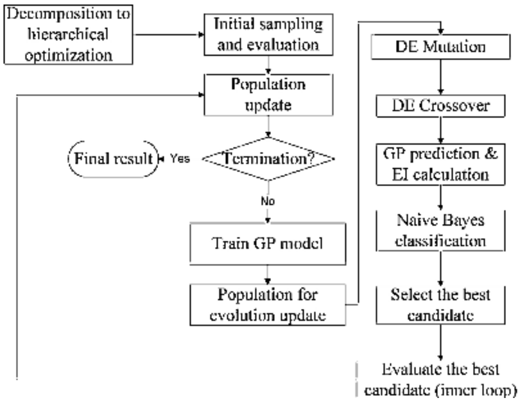
Decomposition to hierarchical optimization
Initial sampling and evaluation
Population update
Final result
Yes
Termination?
No
Train GP model
Population for evolution update
DE Mutation
DE Crossover
GP prediction & EI calculation
Naive Bayes classification
Select the best candidate
Evaluate the best candidate (inner loop)

Fig. 3. Flow diagram of EMLDE.

The construction of SBDE is quite simple. The optimization core is the standard DE algorithm (see Section II-C) with a change of the selection operator. Constraints (e.g., bandwidth $\geq$ 20 GHz) are handled by using the selection rules in [27]. The selection rules are: 1) given two feasible solutions, select the one with the better objective function value; 2) given two infeasible solutions, select the solution with the smaller constraint violation; and 3) if one solution is feasible and the other is not, select the feasible solution. More details are in [23] and [27].

Although the S-parameter simulation is fast, the inner optimization needs to evaluate a full population in each iteration and the optimization needs several iterations. Because the evaluation of different candidate designs in a population is independent of each other in SBDE, we use parallel computation. In our implementation, a 8-core CPU is used. Experiments show that the inner optimization of one candidate transformer design can be finished in 5 to 6 min for the test circuit. This time consumption is quite general, because the time cost of the inner loop is mainly dominated by the S-parameter circuit simulation, which is correlated to the size of the simulated circuit stage. Using the same hardware and the circuit division method based on passive components, the size of each stage often does not vary much even for different amplifiers.

### D. Flow Diagram of EMLDE

The general flow of the high-frequency linear RF amplifier synthesis system is shown in Fig. 2. The flow diagram of the EMLDE algorithm is shown in Fig. 3. ABGPDE and SBDE are included in EMLDE.

The EMLDE algorithm works as presented in Algorithm 1.

In EMLDE, the parameters that need to be set by the user can be classified into the DE parameters, the GP parameters, the number of initial samples, and the parameters for the inner optimization loop. The rules for determining the settings and the robustness for the first three classes are elaborated in [2], and will not be repeated here. For the inner optimization loop, the population size and the number of iterations need to be considered. We suggest setting the population size to be 20. Normally, for an optimization problem with about 10–15

**Algorithm 1** The EMLDE algorithm

```
Step 0: Decompose the problem of optimizing an amplifier stage into hierarchical optimization with outer and inner loops according to the method in Section III.
Step 1: Initialize the parameters, e.g., the DE algorithm parameters.
Step 2: Initialize the population by LHS sampling of the design space and perform EM simulation to the samples (can be done beforehand for one technology). Perform the inner optimization loop for the samples. SBDE, the active component library and the transmission line LUT are used in this step.
Step 3: Update the population $A$ by adding newly generated samples and their performances. In the first iteration, the added samples are from step 2; afterward, they are from step 11. Update the best solution obtained so far.
Step 4: Check if the stopping criterion (e.g., a convergence criterion or a maximum number of iterations) is met. If yes, output the result; otherwise go to step 5.
Step 5: Train the GP surrogate model according to population $A$.
Step 6: Construct the population for evolution (population $B$) as described in Section IV-B2).
Step 7: Use population $B$ and perform the mutation operation according to (9) to obtain each candidate solution's mutant counterpart.
Step 8: Perform the crossover operation between each candidate solution and its corresponding mutant counterpart according to (10) and (11) to obtain each individual's trial individual.
Step 9: Calculate the EI value of all the trial individuals from step 8.
Step 10: Use the population $A$ as the training data, perform naive Bayes classification as described in Section IV-A to all the trial individuals from step 8.
Step 11: Select the individual with the best potential according to Section IV-B2) and evaluate it using the same way as in step 2. Go back to Step 3.
```

variables, which is very typical for one stage of the mm-wave RF amplifier without the inductor or transformer, this setting can often obtain a near-optimal result and is widely applied for real-world problems. A larger value of the population size often has very little or no improvement on the result, but costs more time because the convergence may slow down [19]. The number of iterations, on the other hand, depends on the engineering problem being optimized. For S-parameter optimization in a RF amplifier stage, using 30 iterations is an empirical setting. Through our experiments, we found that the objective function value improvement when using a population of 30 individuals and 100 iterations to different problems (amplifier stages and some mathematical benchmark problems) compared with the above settings is only 2–3%. If the hardware resources are sufficient, such as a 12-core CPU or a cluster, we therefore suggest using 30 individuals in the population and about 30–40 iterations. After all, the designer can always perform tests on a few samples with given fixed passive devices to decide the optimal setting of the number of iterations, which is easy to be done. For an amplifier stage, after finding the best candidate of the transformer or inductor, the user can re-do the specific inner optimization using more iterations. Because only one inner optimization is done this time, it is very cheap. Normally the re-optimization of the inner loop needs less than 10 min, which will not affect the total synthesis time much.

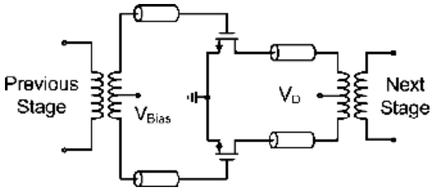

Fig. 4. One stage of the 100 GHz amplifier (the full amplifier needs three such stages).

## V. Experimental Results and Comparisons

In this section, the EMLDE method is demonstrated for the synthesis of a 100 GHz three-stage transformer-coupled fully differential amplifier [22] in a 90 nm CMOS technology. One stage of the circuit configuration is shown in Fig. 4. Using the same configuration but different sizing for each stage, the different stages are cascaded together. The optimization goal is the power gain (S21 at 100 GHz); the constraints are bandwidth $\geq$ 20 GHz and the Rollet stability factors ($K$ factors) [28] $>$ 1. Note that at 100 GHz, design for high gain is difficult due to the limitation of the $f_T$ of the 90 nm technology. Achieving similar performances, [22] used six stages (measurement result), while EMLDE uses three stages. Although simulation result and measurement result cannot be compared directly, the manual design result is a good reference to verify the high solution quality of EMLDE.

The transmission line used is a high-Q slow-wave coplanar transmission line [29]. The differential lines (CPW line) are on the top metal layer and the floating metal strips are on the lower metal layer. For the transformer, the top metal layer is used. All the transistors have the same size to make sure that each stage can drive the next stage. All the transistors have 1 $\mu$m width, 90 nm length and 15 fingers as in [22]. In a cascaded multistage RF amplifier with the same transistor size, the optimal design parameters of the transformers often do not differ much from one stage to the other. As said above, the manual design method of copying the design of one stage to construct the whole amplifier receives less optimal results. Because of this, all the passive components are re-synthesized, but after performing the inner optimization to the initial samplings in the synthesis of the previous stage, we can delete a few samplings which have very bad performances, when synthesizing the next stage. The design variables are as follows. For transformers, the design variables are the inner diameter of the primary inductor (*dinp*), the inner diameter of the secondary inductor (*dins*), the width of the primary inductor (*wp*), the width of the secondary inductor (*ws*), and

TABLE I
DESIGN PARAMETERS AND THEIR RANGES

| Parameters | Lower Bound | Upper Bound |
|---|---|---|
| *dinp*, *dins* ($\mu$m) | 30 | 110 |
| *wp*, *ws* ($\mu$m) | 2 | 10 |
| *sp* ($\mu$m) | 8 | 23 |
| *lw* ($\mu$m) | 1 | 10 |
| *ll* ($\mu$m) | 2 | 80 |
| *ls* ($\mu$m) | 7 | 23 |

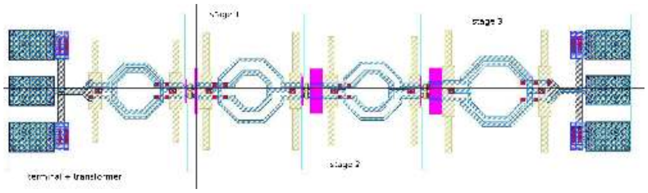

Fig. 5. Amplifier synthesized without machine learning.

the spacing between two ports (*sp*). For transmission lines, the design variables are the metal width (*lw*), the metal length (*ll*), and the spacing between the differential lines (*ls*). Two DC voltages ($V_D$ and $V_{\text{Bias}}$) are included in each stage. The output load impedance is 50 Ω. The ranges for the design variables are given in Table I, which are provided by the designer. There are in total 51 design variables.

The algorithm parameter settings are as follows. In ABGPDE, the parameters are the same as in [2], where the robustness has been analyzed. For the GP model used in ABGPDE, the DACE toolbox [30] is used. For SBDE, we set the population size to 20 and the number of iterations is 20. The purpose of this setting is to call for a good balance between the solution quality and the efficiency of the inner optimization. EMLDE stops when the performance cannot be improved for 20 consecutive generations or when the number of outer iterations reaches 90 (including initial points). The examples are run on a PC with Intel 2.66 GHz dual Xeon 2×6 core-CPU (only eight cores are used) under the Linux operating system. All the time consumptions mentioned in the experiments are clock time. Advanced design system momentum is used as the EM simulator. Note that each EM simulation is parallelized automatically by momentum using the eight cores in all the methods. Synopsys HSPICE is used as the circuit simulator with S-parameter models.

We synthesize the amplifier stage by stage, from the output stage forward to the input stage. For the current stage being optimized, the S-parameter models of the passive components (transformer, transmission lines) are separated for HSPICE simulation, while the already synthesized stage is described by a single S-parameter model integrating all the passive components. For example, when optimizing stage 2, the transformer and the transmission lines have their own S-parameter models to enter the HSPICE simulation. For stage 3, which is already synthesized, the pad, transformer and transmission lines are connected together to perform a EM simulation, whose result will be used for stage 2. Because the amplifier includes four parts (see Fig. 5) to synthesize, and the matching impedances of each part are different, four test problems are included in this example. The performance of the whole amplifier is affected by all of the four test problems, which shows the robustness of EMLDE.

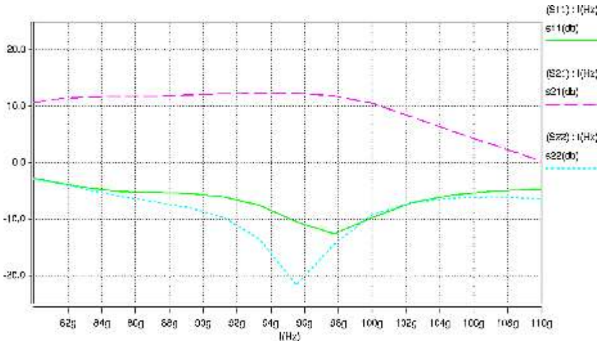

Fig. 6. S-parameters curve from 80 GHz to 110 GHz (experiment 1).

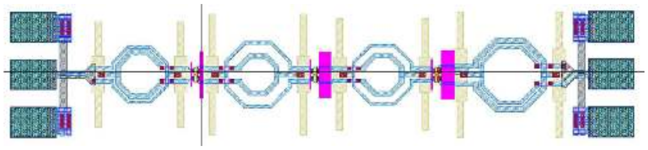

Fig. 7. Amplifier synthesized by EMLDE.

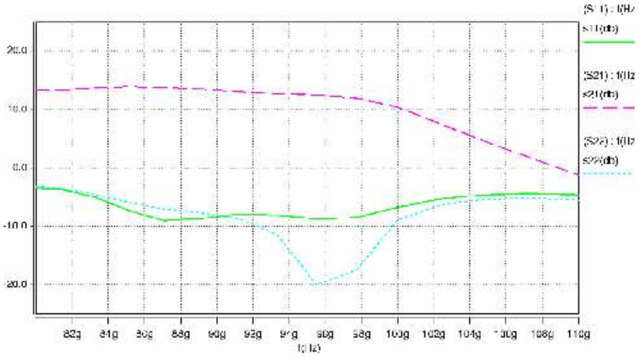

Fig. 8. S-parameter curve from 80 GHz to 110 GHz (experiment 2).

### *A. Three-Stage Linear Amplifier Synthesis*

We first use the method of the DE algorithm with EM simulation for transformers as the performance evaluation. No machine learning method is used. This method can provide the best result, which can serve as a reference result, but is of course very CPU time expensive. Parallel computation of the HSPICE simulations is not included. The synthesized circuit is shown in Fig. 5 and the simulation results are shown in Fig. 6. The power gain is 10.53 dB and the time consumption is about nine days.

Then, we use the proposed EMLDE-based synthesis system. The synthesized circuit is shown in Fig. 7 and the simulation results are shown in Fig. 8. The power gain is 10.41 dB and the time cost is only 25 h. The constraints are satisfied for both methods.

It can be seen that the proposed high-frequency RF linear amplifier synthesis system can achieve a comparable result compared with directly using the DE algorithm and EM simulation, which is the comparison benchmark. We can also see the high solution quality from the mm-wave frequency RF IC design aspect. It is well known by designers that achieving 3 dB power gain per stage requires very good matching beyond 60 GHz, and that the higher the working frequency, the more difficult it is to achieve high gain. The result shows that the average power gain of each stage reaches nearly 3.5 dB in the synthesized amplifier at 100 GHz considering the loss of the passive components and the pads. In terms of computational efficiency, about nine times speed enhancement is achieved. The time cost of 25 h is very reasonable for practical use.

In this synthesis, a total of 48 EM simulations are used (excluding the 49 initial sampling points which can be done beforehand in a given technology). The time spent on EM simulations is 2.5 h. The inner loop optimization costs 22.7 h. When directly using the DE algorithm and EM simulation, nearly 4000 EM simulations are needed. It can be seen that EMLDE decreases the number of expensive EM simulations by about 80 times. Although there are much more circuit simulations in EMLDE, the linear circuit simulation is cheap. We can also conclude that the more complex the key passive components, which need more EM simulation time, the higher the advantage of EMLDE.

### B. Comparisons

To show the advantages of the proposed ABGPDE algorithm, a comparison with the prescreening method used in standard EGO [15] is performed. Both methods use GP machine learning and the EI prescreening method. DE is selected as the optimization core to both methods. The differences between ABGPDE and the comparison method can be summarized into two main points: 1) ABGPDE uses the naive Bayes classifier to help the EI prescreening in order to make the selection of the promising candidate more reliably; and 2) the population is adjusted to increase the amount of beneficial information in each iteration. The purpose is to enhance the efficiency of generating promising candidates.

The test problem is the synthesis of the most difficult stage (the third stage) in the amplifier to show the different ability of the two algorithms. Because the output pad is included, the matching is more difficult and the gain is lower than a middle stage (e.g., second stage) due to the parasitics and loss caused by the pad. In addition, because this stage is handled first in synthesis, all the initial samples need to perform the inner optimization. The design variables include the parameters from the one transformer, the two transmission lines and the two DC voltages. In this experiment, the maximum number of iterations of the outer optimization loop was set to 150 (including the 49 initial samples). Ten runs are performed for each method. The results are summarized in Table II, where N(inner) is the number of inner optimization loops when obtaining the best solution, including the inner optimization of the 49 initial samples.

From Table II, we can see that the naive Bayes classifier and the adaptive population generation method in ABGPDE improve both the solution quality and the efficiency clearly.

TABLE II
ANALYSIS OF ABGPDE

| | ABGPDE | Comparison Method |
|---|---|---|
| Best gain | 2.41 dB | 2.24 dB |
| Worst gain | 2.12 dB | 1.87 dB |
| Average gain | 2.28 dB | 2.06 dB |
| Best N(inner) | 61 | 70 |
| Worst N(inner) | 98 | 150 |
| Average N(inner) | 75 | 109 |
| Average clock time | 8.0 h | 12.8 h |

We also compared with the method of directly using EI prescreening (but not using naive Bayes classifier and adaptive population generation) without an inner loop, i.e., the 13 variables plain optimization problem. Because the inner loop does not exist in the plain optimization problem, we set the run time to be 13 h (almost the average clock time of a run of the comparison method in Table II) and look at the results. The best result in five runs is only 1.04 dB. This confirms the dimensionality problem for Gaussian process surrogate model-based optimization.

### C. Test and Comparison Framework for RF Amplifiers

There are three main issues which are interesting for circuit design automation researchers. They are the comparison to other methods, the examination of newly developed algorithms and their robustness. However, unlike for low-frequency analog ICs, it is often not easy to do experiments for high-frequency ICs. The reason is that the computation is very expensive; even 25 h in this paper is not a short time compared with 10–20 min for low-frequency analog circuit sizing. Testing new algorithms and performing more runs to test the robustness then becomes very time consuming. For comparisons, because the published RF circuits often use different technologies, which are critical in mm-wave circuit design, it is also difficult to make a fair comparison between publications. To address these problems, instead of adding a new circuit example, we present a test and comparison framework by reconstructing mathematical benchmark problems in the EC field for RF IC designers.

In the EC field, there are a bunch of mathematical benchmark problems with different properties, such as having many local optimal points, having discontinuous and nondifferentiable. More details of the functions and the corresponding search domains can be found in [31] and [32]. We select unconstrained single objective optimization benchmark problems to construct the test problems for RF IC synthesis. An example is provided in the following and more examples can be constructed using the same way.

The benchmark problem selected is the Ackley function (minimization). The objective function is shown in

$$f(x)= -20 \cdot \exp\left(-0.2 \cdot \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + \exp(1). \tag{15}$$

The search domain of this test function is $-32.768 \le x_i \le 32.768$ $(i = 1, \ldots, n)$. The global optimal point is that all the $x_i$ are 0 and the corresponding objective function value is 0.

Because this problem is scalable, we choose $n = 17$ to mimic a RF amplifier stage. Five of the design variables are used to mimic the passive component design. Therefore, the new problem is shown in

$$f(x) = Ackley_{13\ \mathrm{dim}}(x_1, \ldots, x_{12}, Ackley_{5\ \mathrm{dim}}(x_{13}, \ldots, x_{17}))$$
$$\text{subject to}: \sum_{i=1}^{17} x_i \leq 10. \tag{16}$$

The constraint is reasonable. We do not suggest using constrained optimization benchmark problems, because the constraints in those problems are often very severe to test the constraint handling ability of an algorithm, which is not the case for RF IC synthesis. When the 5-D Ackley function (passive component) is fully optimized, the objective function is 0, which is just the global optimal point of the 13rd design variable of the 13-D Ackley function (the amplifier stage) in the decision space. We consider the 5-D Ackley function to be expensive (in reality it is not) and only require a limited number of simulations.

With the constructed problems, the method to judge if an obtained solution is optimal is interesting for RF IC synthesis researchers. Because many benchmark problems are often more difficult than real-world problems, besides comparing with the known global optimal solution, the information on the extent of difficulty of the problem is also necessary. Here, we take the Ackley test function as the example. The Ackley function has numerous local optimal points, and available state-of-the-art surrogate model assisted evolutionary algorithms often get an optimal function value of about 4 to 6 (e.g., [18]) in about a thousand expensive function evaluations for the 20-D Ackley function, compared with the global optimal function value 0. Although we cannot compare with these methods directly, because they do not assume the hierarchical structure which appears in linear RF IC synthesis, those results are an indication of the difficulty of the test problem. For the Ackley function, synthesis algorithms obtaining an average function value smaller than 1 using the obtained optimal solutions should be good for application.

We run the EMLDE algorithm to this problem for 20 times. Because this problem has numerous local optimal points, we replace the last 10% low-quality candidates by candidates with beneficial information to maintain the diversity and lower the threshold of classification to avoid beneficial candidates being removed. The number of iterations in the inner loop is set to 350. As stated above, the number of runs in the inner loop is decided by the kind of engineering problem. The other settings are the same as Section V-A. The best, worst, and average results are 0.0452, 3.95, and 0.67, respectively. In the 20 runs, the results of 16 runs are below 1, 15 of them are below 0.5, and a few of them are below 0.1. Hence, EMLDE can obtain a good solution in 80% probability, which is quite robust, especially in surrogate model assisted evolutionary algorithms.

The proposed framework has three advantages.

1) The experiment is fast. Therefore, it is easy to have sufficient runs to test the robustness, test new methods, and do comparisons between different methods.
2) The comparison is not limited by the technology used, and allows us to compare different methods fairly.
3) The global optimal solution and the extent of difficulty of the problem is known, which provides a criterion to judge the quality of the method. For the synthesis of mm-wave circuits, it is difficult to know the global optimal point, since there does not exist any viable method till now. But using this framework, we can still evaluate new methods using the available information. The limitation of this framework is that the parameter settings cannot be directly used to RF IC synthesis algorithms, because these mathematical benchmark problems are normally more difficult than real-world problems.

## VI. Conclusion

In this paper, the first synthesis method, EMLDE, for mm-wave frequency linear RF amplifiers was presented. The core ideas were the decomposition method, which reformulates the problem to a hierarchical structure, and the proposed ABG-PDE algorithm to solve low-dimensional but more complex expensive optimization problems. The parallel computation technique also contributed positively to the EMLDE method. EMLDE can provide results that are comparable to directly using a global optimization algorithm with EM simulations as performance evaluation, which is the best framework in terms of solution quality, but at far lower computational cost. As an example, we showed a 100 GHz three-stage linear amplifier successfully being synthesized with more than 10 dB gain, 20 GHz bandwidth, while only using 25 h clock time (nine times speedup). The goals of high optimization ability, high efficiency, and high generality were therefore met. In addition, a test and comparison framework for future use by RF IC design automation researchers was presented.

## Acknowledgment

The authors would like to sincerely thank B. Machiels and B. Su for valuable discussions.

## References

[1] S. Ohmori, Y. Yamao, and N. Nakajima, "The future generations of mobile communications based on broadband access technologies," *IEEE Commun. Mag.*, vol. 38, no. 12, pp. 134–142, Dec. 2000.

[2] B. Liu, D. Zhao, P. Reynaert, and G. G. E. Gielen, "Synthesis of integrated passive components for high-frequency RF ICs based on evolutionary computation and machine learning techniques," *IEEE Trans. Comput.-Aided Des. Integr. Circuit Syst.*, vol. 30, no. 10, pp. 1458–1468, Oct. 2011.

[3] D. J. Allstot, K. Choi, and J. Park, *Parasitic-Aware Optimization of CMOS RF Circuits*. Dordrecht, The Netherlands: Kluwer Academic, 2003.

[4] K. Choi and D. J. Allstot, "Parasitic-aware design and optimization of a CMOS RF power amplifier," *IEEE Trans. Circuits Syst. I*, vol. 53, no. 1, pp. 16–25, Jan. 2006.

[5] G. Tulunay, "A synthesis tool for CMOS RF low-noise amplifiers," *IEEE Trans. Comput.-Aided Des. Integr. Circuit Syst.*, vol. 27, no. 5, pp. 977–982, May 2008.

[6] J. Ramos, K. Francken, G. G. E. Gielen, and M. S. J. Steyaert, "An efficient, fully parasitic-aware power amplifier design automation tool," *IEEE Trans. Circuits Syst. I*, vol. 52, no. 8, pp. 1526–1534, Aug. 2005.

[7] A. Agarwal and R. Vemuri, "Layout-aware RF circuit synthesis driven by worst case parasitic corners," in *Proc. ICCD*, Oct. 2005, pp. 444–449.

[8] G. Zhang, A. Dengi, R. A. Rohrer, R. A. Rutenbar, and L. R. Carley, "A synthesis flow toward fast parasitic closure for radio-frequency integrated circuits," in *Proc. DAC*, Jul. 2004, pp. 155–158.

[9] C. R. C. De Ranter, G. Van der Plas, M. S. J. Steyaert, G. G. E. Gielen, and W. M. C. Sansen, "CYCLONE: Automated design and layout of RF LC-oscillators," *IEEE Trans. Comput.-Aided Des. Integr. Circuit Syst.*, vol. 21, no. 10, pp. 1161–1170, Oct. 2002.

[10] M. Chu and D. J. Allstot, "Elitist nondominated sorting genetic algorithm based RF IC optimizer," *IEEE Trans. Circuits Syst. I*, vol. 52, no. 3, pp. 535–545, Mar. 2005.

[11] J. Park, K. Choi, and D. J. Allstot, "Parasitic-aware design and optimization of a fully integrated CMOS wideband amplifier," in *Proc. ASP-DAC*, 2003, pp. 904–907.

[12] B. Liu, F. V. Fernández, G. Gielen, R. Castro-López, and E. Roca, "A memetic approach to the automatic design of high performance analog integrated circuits," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 14, no. 3, p. 42, 2009.

[13] M. Stein, "Large sample properties of simulations using Latin hypercube sampling," *Technometrics*, vol. 29, no. 2, pp. 143–151, 1987.

[14] S. K. Mandal, S. Sural, and A. Patra, "ANN- and PSO-based synthesis of on-chip spiral inductors for RF ICs," *IEEE Trans. Comput.-Aided Des. Integr. Circuit Syst.*, vol. 27, no. 1, pp. 188–192, Jan. 2008.

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

[16] P. D. Wasserman, *Neural Computing: Theory and Practice*. New York: Van Nostrand Reinhold, 1988.

[17] T. J. Santner, B. J. Williams, and W. Notz, *The Design and Analysis of Computer Experiments*. Berlin, Germany: Springer, 2003.

[18] M. T. M. Emmerich, K. C. Giannakoglou, and B. Naujoks, "Single- and multiobjective evolutionary optimization assisted by Gaussian random field metamodels," *IEEE Trans. Evol. Comput.*, vol. 10, no. 4, pp. 421–439, Aug. 2006.

[19] K. V. Price, R. M. Storn, and J. A. Lampinen, *Differential Evolution. A Practical Approach to Global Optimization*. Berlin, Germany: Springer, 2005.

[20] D. Jones, "A taxonomy of global optimization methods based on response surfaces," *J. Global Optimization*, vol. 21, no. 4, pp. 345–383, 2001.

[21] Q. Zhang, W. Liu, E. Tsang, and B. Virginas, "Expensive multiobjective optimization by MOEA/D with Gaussian process model," *IEEE Trans. Evol. Comput.*, vol. 14, no. 3, pp. 456–474, Jun. 2010.

[22] N. Deferm and P. Reynaert, "A 100 GHz transformer-coupled fully differential amplifier in 90 nm CMOS," in *Proc. RFIC*, May 2010, pp. 359–362.

[23] B. Liu, F. V. Fernandez, 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 Des. Integr. Circuits Syst.*, vol. 30, no. 6, pp. 793–805, Jun. 2011.

[24] D. Larose, *Data Mining Methods and Models*. New York: Wiley, 2006.

[25] R. Caruana and A. N. Mizil, "An empirical comparison of supervised learning algorithms," in *Proc. Int. Conf. Mach. Learning*, 2006, pp. 161–168.

[26] Z. Michalewicz, "A survey of constraint handling techniques in evolutionary computation methods," in *Proc. 4th Annu. Conf. Evol. Programming*, 1995, pp. 135–155.

[27] K. Deb, "An efficient constraint handling method for genetic algorithm," *Comput. Meth. Appl. Mech. Eng.*, vol. 186, nos. 2–4, pp. 311–338, 2000.

[28] J. Rollett, "Stability and power-gain invariants of linear twoports," *IRE Trans. Circuit Theory*, vol. 9, no. 1, pp. 29–32, Mar. 1962.

[29] D. Kaddour, H. Issa, A. L. Franc, N. Corrao, E. Pistono, F. Podevin, J. M. Fournier, J. M. Duchamp, and P. Ferrari, "High-Q slow-wave coplanar transmission lines on 0.35 *mu*m CMOS process," *IEEE Microw. Wirel. Compon. Lett.*, vol. 19, no. 9, pp. 542–544, Sep. 2009.

[30] S. N. Lophaven, H. B. Nielsen, and J. Sondergaard, "DACE: A MATLAB Kriging Toolbox," Tech. Univ. Denmark, Copenhagen, Denmark, Tech. Rep., 2002.

[31] C. A. Floudas, P. M. Pardalos, C. Adjiman, W. R. Esposito, Z. H. Gümüs, S. T. Harding, J. L. Klepeis, C. A. Meyer, and C. A. Schweiger, *Handbook of Test Problems in Local and Global Optimization*. Berlin, Germany: Springer, 1999.

[32] T. B. Beielstein, *Experimental Research in Evolutionary Computation: The New Experimentalism*. Berlin, Germany: Springer, 2006.

**Bo Liu** was born in Beijing, China, on September 23, 1984. He received the B.S. degree from Tsinghua University, Beijing, in 2008. Since 2008, he has been a Ph.D. candidate and a Research Assistant with the MICAS Laboratories, Katholieke Universiteit Leuven, Leuven, Belgium, under the supervision of Dr. G. Gielen.

He has authored or co-authored more than 20 papers in international journals and conference proceedings. He is currently a Technical Consultant with Accelicon Technologies, Inc., Cupertino, CA. His current research interests include evolutionary computation, machine learning, fuzzy logic, and design automation methodologies of analog and radio frequency integrated circuits.

Mr. Liu is a Reviewer in artificial intelligence and analog design automation fields, such as the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, *Information Sciences* (Elsevier), and *Integration, the VLSI Journal* (Elsevier). He is also a Book Reviewer of Elsevier and Bentham Science.

**Noël Deferm** was born in Diest, Belgium, in December 1985. He graduated from Katholieke Universiteit Leuven, Leuven, Belgium, in 2008. He received the Masters degree in electrical engineering. Currently, he is pursuing the Ph.D. degree in mm-wave CMOS integrated circuit design with the MICAS Research Group, Department of Electrical Engineering, Katholieke Universiteit Leuven.

**Dixian Zhao** received the B.Sc. degree in microelectronics from Fudan University, Shanghai, China, in 2006, and the M.Sc. degree in microelectronics from the Delft University of Technology, Delft, The Netherlands, in 2009. Currently, he is pursuing the Ph.D. degree with Katholieke Universiteit Leuven, Leuven, Belgium.

From 2005 to 2007, he was with the Auto-ID Laboratory, Fudan University, designing the nonvolatile memory for passive radio-frequency identification tags. From 2008 to 2009, he was an Intern with Philips Research, Eindhoven, The Netherlands, where he designed a 60 GHz beamforming transmitter for the presence of detection radar. From 2009 to 2010, he was with the Delft University of Technology, where he developed the 94 GHz wideband receiver for imaging radar. His current research interests include radio frequency and millimeter-wave integrated circuit design for wireless communications.

**Patrick Reynaert** was born in Wilrijk, Belgium, in 1976. He received the Masters of Industrial Sciences degree in electronics (ing.) from the Karel de Grote Hogeschool, Antwerpen, Belgium, in 1998, and both the Masters of Electrical Engineering (ir.) and the Ph.D. degrees in engineering science (dr.) from Katholieke Universiteit Leuven (K. U. Leuven), Leuven, Belgium, in 2001 and 2006, respectively.

From 2001 to 2006, he was a Teaching and Research Assistant with the MICAS Research Group, Department of Electrical Engineering (ESAT), K. U. Leuven. From 2006 to 2007, he was a Post-Doctoral Researcher with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley. In 2007, he was a Visiting Researcher with Infineon, Villach, Austria. Since October 2007, he has been an Associate Professor with ESAT, K. U. Leuven, and a Staff Member with the ESAT-MICAS Research Group. His current research interests include CMOS power amplifiers and mm-wave CMOS integrated circuits.

**Georges G. E. Gielen** (S'87–M'92–SM'99–F'02) received the M.Sc. and Ph.D. degrees in electrical engineering from Katholieke Universiteit Leuven, Leuven, Belgium, in 1986 and 1990, respectively.

In 1990, he was a Post-Doctoral Research Assistant and a Visiting Lecturer with the Department of Electrical Engineering and Computer Science, University of California, Berkeley. In 1993, he was an Assistant Professor with Katholieke Universiteit Leuven, where he was promoted to a Full Professor in 2000. He has authored or co-authored two books and more than 400 papers in edited books, international journals, and conference proceedings. His current research interests include the design of analog and mixed-signal integrated circuits, and especially analog and mixed-signal computer-aided design tools and design automation (modeling, simulation and symbolic analysis, analog synthesis, analog layout generation, analog, and mixed-signal testing).

Dr. Gielen received the 1995 Best Paper Award in the John Wiley *International Journal on Circuit Theory and Applications*, and was the 1997 Laureate of the Belgian Royal Academy on Sciences, Literature and Arts in the Discipline of Engineering. He received the 2000 Alcatel Award from the Belgian National Fund of Scientific Research for his innovative research in telecommunications, and won the DATE 2004 Best Paper Award. He has served as an Elected Member of the Board of Governors of the IEEE Circuits and Systems (CAS) Society and as the Chairman of the IEEE Benelux CAS Chapter. He served as the President of the IEEE CAS Society in 2005.