Harvesting Design Knowledge From the Internet: High-Dimensional Performance Tradeoff Modeling for Large-Scale Analog Circuits

Jun Tao, *Member, IEEE*, Changhai Liao, Xuan Zeng, *Member, IEEE*, and Xin Li, *Senior Member, IEEE*

***Abstract*—Efficiently optimizing large-scale, complex analog systems requires to know the performance tradeoffs for various analog circuit blocks. In this paper, we propose a radically new approach for analog performance tradeoff modeling. Our key idea is to broadly search the rich design knowledge from the Internet, and then mathematically encode the knowledge as high-dimensional performance tradeoff curves that are referred to as Pareto fronts in the literature. Toward this goal, several novel numerical algorithms, such as sparse regression and semi-infinite programming, are developed in order to construct the high-dimensional Pareto front model while guaranteeing its monotonicity. Our numerical examples demonstrate that the proposed modeling technique can accurately capture the high-dimensional Pareto fronts for large-scale analog systems (e.g., analog-to-digital converter) while most traditional methods are limited to low-dimensional Pareto front modeling of small circuit blocks without considering layout parasitics and manufacturing nonidealities.**

***Index Terms*—Analog circuit, Pareto front, performance tradeoff.**

I. Introduction

As AN indispensable portion of modern integrated systems, analog circuit often becomes the major bottleneck that limits system performance, product yield, and time to market. During the past two decades, numerous optimization algorithms have been developed to facilitate efficient analog circuit design [1]–[4]. The objective of these algorithms is to automatically determine the optimal device sizes (e.g., transistor width) based on numerical simulations and/or design equations. Analog optimization techniques have been successfully incorporated into commercial computer-aided design (CAD) tools and applied to synthesize a large number of analog circuit blocks such as operational amplifiers, and voltage-controlled oscillators (VCOs).

Manuscript received December 20, 2014; revised March 26, 2015 and May 20, 2015; accepted June 9, 2015. Date of publication June 23, 2015; date of current version December 18, 2015. This work was supported in part by the National Natural Science Foundation of China under Project 61376041, Project 61125401, and Project 61376040, in part by the National Basic Research Program of China under Grant 2011CB309701, and in part by the Shanghai Science and Technology Committee Project under 13XD1401100. This paper was recommended by Associate Editor S.-C. Chang. *(Corresponding author: Xuan Zeng.)*

J. Tao, C. Liao, and X. Zeng are with the State Key Laboratory of ASIC & System, Department of Microelectronics, Fudan University, Shanghai 200433, China (e-mail: taojun@fudan.edu.cn; xzeng@fudan.edu.cn).

X. Li is with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA (e-mail: xinli@ece.cmu.edu).

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.2015.2449240

While analog optimization has made remarkable advance at block level, efficiently optimizing large-scale, complex analog systems remains a challenging task. Typically, a hierarchical approach is adopted to attack this system-level optimization problem [5]–[17]. Namely, analog optimization is first performed to extract the performance tradeoffs, referred to as Pareto fronts, at block level. Next, system-level optimization is further pursued based on these tradeoff models. To make such a hierarchical optimization approach of practical utility, Pareto front modeling is one of the key components. While there have been a large body of techniques proposed for Pareto front modeling in [5]–[17], they are often limited to small-scale (i.e., small circuit blocks) and/or low-dimensional (i.e., 2–3 performance metrics) problems covering few circuit architectures and technology nodes only. The capability of today's Pareto front modeling is heavily constrained by the computational resource of running numerical simulations and/or the human resource of setting up design equations.

For instance, consider analog-to-digital converter (ADC) as an example. To the best of our knowledge, there is no existing technique that can extract the Pareto fronts for ADC to cover all different performance metrics (e.g., area, power, speed, and signal-to-noise distortion ratio (SNDR)), circuit architectures (e.g., flash ADC, pipeline ADC, and successive approximation ADC), and technology nodes (e.g., 22 nm, 32 nm, and 45 nm). Such a Pareto front model, however, is extremely critical, when a system-level designer makes important design choices (e.g., which circuit architecture and technology node should be chosen) to implement a given product.

The technical challenge here is not about Pareto front modeling itself: sophisticated, high-dimensional Pareto front models have been created in other fields such as microeconomics [18]. Instead, the fundamental challenge is how to collect the data that are required for Pareto front modeling. Traditionally, simulation- and/or equation-based optimizations have been applied to generate the data. Such an approach relying on circuit optimization is not scalable, thereby making large-scale, high-dimensional modeling problem intractable.

In this paper, we rethink Pareto front modeling from a completely different perspective. Our proposed work is motivated by the fact that enormous design data are available on the Internet (e.g., published papers and data sheets). For example, IEEE Xplore carries more than 25 000 papers with silicon measurement data published at the Journal of Solid-State Circuits (JSSC) or the International Solid-State Circuit Conference (ISSCC). These data include the performance metrics for a large number of silicon-proved designs. They contain valuable design information and, most importantly, are almost free to the public. Analog designers have been continuously reading these online resources and learning design knowledge from the Internet. Can a CAD tool also exploit the publically available design information and utilize it in an analog design flow?

To explore the rich design knowledge from the Internet, two important techniques must be developed.

1) *Knowledge Search and Data Harvest:* Instead of looking for the online design knowledge manually, a search engine must be developed to automatically process the information on various Web sites and extract the design data. In the machine learning community, text mining is a well-established technique to systematically analyze the structured-data from the Internet [19]. It perfectly fits our need and several commercial off-the-shelf tools can be directly used. Hence, we will not further discuss the topic of automatic data search in this paper. More details can be found from [19].
2) *Knowledge Discovery and Pareto Front Modeling:* Once the big data set of circuit performance is available, we need to further extract a compact expression (i.e., the Pareto front model) to represent the performance tradeoffs. Mathematically, it requires us to approximate the high-dimensional Pareto front by an analytical function. In this paper, we will mainly focus on this Pareto front modeling problem and develop novel numerical algorithms to make it computationally efficient.

In particular, we propose to adopt the sparse regression technique [20]–[22] to generate high-dimensional Pareto fronts. Here, sparse regression is chosen, because the shape of a Pareto front is not known in advance. Hence, it is extremely difficult, if not impossible, to predetermine a set of basis functions to approximate the Pareto front of interest. Sparse regression is a statistical technique that adaptively selects the appropriate basis functions from a large pool of possible candidates. Such a capability of basis function selection is especially important for high-dimensional modeling where there are numerous basis functions to span the high-dimensional performance space. For this reason, sparse regression serves as an efficient dimension reduction technique to make our high-dimensional Pareto front modeling problem tractable.

Furthermore, the Pareto front of an analog circuit must be monotonic inside the feasible performance region. To guarantee such monotonicity, we propose to further constrain the gradient of the Pareto front, resulting in a semi-infinite programming problem for Pareto front modeling. The grid discretization method [23] is then adopted to efficiently solve the aforementioned semi-infinite programming problem and determine the Pareto front of interest. As will be demonstrated by the numerical experiments in Section V, our proposed sparse regression algorithm based on grid discretization can construct high-dimensional Pareto fronts both accurately and efficiently while maintaining their monotonicity.

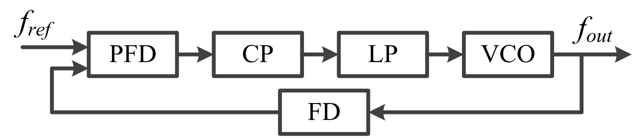


Fig. 1. Simplified block diagram of a PLL includes five circuit blocks: 1) PFD; 2) CP; 3) LP; 4) VCO; and 5) FD [7].

The remainder of this paper is organized as follows. In Section II, we first briefly review the background of system-level optimization and Pareto front modeling for analog circuits. Next, we derive our mathematical formulation for Pareto front modeling in Section III, and then develop several numerical algorithms to solve the modeling problem in Section IV. The efficacy of our proposed technique is demonstrated by several circuit examples in Section V. Finally, we conclude in Section VI.

II. Background

A. System-Level Optimization

To efficiently optimize large-scale analog systems, a hierarchical approach is often taken [7]–[14]. The basic idea is to partition the complex system into several circuit blocks. Next, the system-level performance specifications are mapped to a set of block-level specifications by constraint mapping [14]. During this process, behavioral models are often built for all circuit blocks so that the system-level performance metrics can be efficiently evaluated by behavioral simulations [24].

Consider the phase-locked loop (PLL) in Fig. 1 as an example, its system-level performance metrics include tracking range, locking time, jitter, power, etc. The PLL can be partitioned into five circuit blocks: 1) phase frequency detector (PFD); 2) charge pump (CP); 3) loop filter (LP); 4) VCO; and 5) frequency divider (FD) [7]. Each circuit block has its own performance metrics. For system-level optimization, each circuit block should be represented as a parameterized behavioral model where the block-level performance metrics are considered as the tunable parameters. For instance, the performance metrics of the VCO include center frequency, gain, jitter, power, etc. Its parameterized behavioral model should include tunable center frequency, gain, jitter, etc. With these parameterized behavioral models, we can efficiently run system-level simulations and then decide the optimal performance specifications for all circuit blocks.

B. Pareto Front Modeling

Since the aforementioned system-level optimization considers the block-level performance metrics as optimization variables, we must carefully model the block-level performance tradeoffs before running system-level optimization. For instance, there is a well-known tradeoff between jitter and power for VCO [7]. When designing a VCO, we have to increase its power in order to reduce the jitter. These block-level performance tradeoffs define the fundamental performance limitations of the given analog circuit. Hence, it is crucial to accurately set up these block-level performance constraints for system-level optimization. Otherwise, without taking into account these constraints, the system-level optimization may converge to a set of block-level performance specifications that are not feasible and, consequently, cannot be achieved by circuit-level implementation.

Toward this goal, the concept of feasible region has been adopted to define the achievable performance space for a given analog circuit block [5]–[17]. Without loss of generality, we use a vector $\mathbf{p} = [p_1, p_2, \ldots, p_M] \in \Re^M$ to denote the M performance metrics of a circuit block. The feasible region includes all possible values of $\mathbf{p}$ that are achievable. Each point inside the feasible region is referred to as a feasible point. A Pareto front represents a set of feasible points on the boundary of the feasible region. In other words, a Pareto front describes the block-level performance constraint that is required for system-level optimization.

A feasible point on the Pareto front is referred to as the Pareto optimal point. It cannot be dominated by any other feasible point in the feasible region. Namely, if a point $\mathbf{p}_A$ is Pareto optimal, there exists no other feasible point $\mathbf{p}_B$ that is superior over $\mathbf{p}_A$

$$\mathbf{p}_A \neq \mathbf{p}_B \text{ and } p_{A,m} \leq p_{B,m} \ (m = 1, 2, \ldots, M) \tag{1}$$

where $p_{A,m}$ and $p_{B,m}$ denote the mth element of the vectors $\mathbf{p}_A$ and $\mathbf{p}_B$, respectively. In (1), we assume that a performance value (say, $p_{B,m}$) dominates another performance value (say, $p_{A,m}$), if $p_{B,m}$ is greater than $p_{A,m}$. If this assumption does not hold for a performance metric, a simple linear transformation can be applied to make the notation consistent. Considering the power of an OpAmp as an example, we always prefer a small power value, instead of a large power value. In this case, we can redefine -power as the performance of interest. Since a large value of -power (i.e., a small value of power) is preferred, the performance definition of -power now becomes consistent with the notation in (1).

Most Pareto front modeling methods in the literature attempt to solve the following multiobjective optimization problem [5]–[11], [14]–[16]:

$$\begin{aligned} \underset{x}{\text{maximize}} \quad & \mathbf{p}(\mathbf{x}) = \begin{bmatrix} p_1(\mathbf{x}) & p_2(\mathbf{x}) & \cdots & p_M(\mathbf{x}) \end{bmatrix}^T \\ \text{subject to} \quad & \mathbf{c}(\mathbf{x}) \geq \mathbf{0} \end{aligned} \tag{2}$$

where $\mathbf{x}$ includes all design variables of a circuit block (e.g., transistor sizes of an OpAmp). The performance metrics $\mathbf{p}$ are functions of the design variables $\mathbf{x}$ and they can be obtained by transistor-level simulations. The constraints $\mathbf{c}(\mathbf{x}) \geq \mathbf{0}$ in (2) are imposed to guarantee the proper operation of the circuit. For instance, all CMOS transistors in an OpAmp must stay in the saturation region.

Both stochastic techniques [6], [11] and deterministic methods [5], [16] have been developed to find a number of Pareto optimal points by solving (2). Next, polynomial fitting is often applied to approximate the M-D Pareto front $p_M = f_M(p_1, p_2, \ldots, p_{M-1})$ [6], [7]. Unfortunately, since most existing Pareto front modeling methods rely on multiobjective programming based on transistor-level simulations, they share several critical limitations.

1) Layout parasitics and manufacturing nonidealities are often ignored by the existing methods. In practice, it is unlikely to design an analog circuit whose presilicon simulation results perfectly match the postsilicon measurement data, especially for radio frequency circuits operating at extremely high frequencies. Therefore, the Pareto fronts obtained by the conventional methods based on presilicon transistor-level simulations may not accurately capture the postsilicon reality in practice.
2) Transistor-level simulations are often expensive and, hence, the conventional Pareto front modeling methods are limited to low-dimensional problems and/or small-scale circuits. For instance, the conventional methods typically require more than 1000 transistor-level simulations to build a 2-D Pareto front [5], [6]. It is extremely difficult, if not impossible, to run such a large number of simulations for a large-scale analog circuit (e.g., ADC), since each simulation may take a few days or even months to finish.

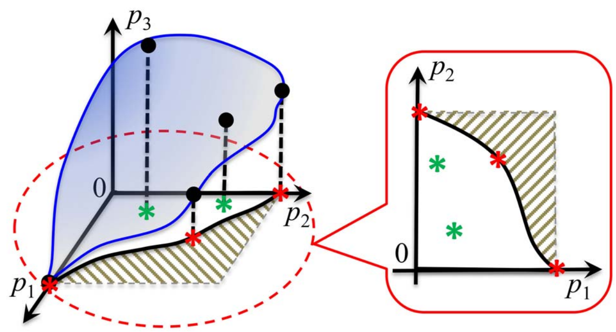


Fig. 2. Simple example of 3-D Pareto front is shown for illustration purpose. Left: 3-D Pareto front is plotted for three performance metrics p_1, p_2, and p_3. There are five Pareto optimal points denoted as the black dots in this 3-D performance space. Right: 3-D Pareto front is projected to the 2-D plane defined by two performance metrics p_1 and p_2. There are only three Pareto optimal points denoted as the red stars in this 2-D performance space.

For these reasons, how to accurately and efficiently build high-dimensional Pareto fronts for large-scale analog circuits remains an open question in the literature. In what follows, we will propose a new Pareto front modeling method to address this grand challenge.

III. Pareto Front Modeling

Mathematically expressing a Pareto front is not trivial, because it cannot be simply specified by a single nonlinear function. To understand the reason, we consider the 3-D Pareto front in Fig. 2. At the first glance, it seems that the 3-D Pareto front can be fully specified by a 2-D nonlinear function $p_3 = f_3(p_1, p_2)$. However, such a simple description is incomplete. As shown in Fig. 2 where the 3-D Pareto front is projected onto the 2-D plane defined by two performance metrics p_1 and p_2, there is an important tradeoff between p_1 and p_2, but it is not captured by the nonlinear function $p_3 = f_3(p_1, p_2)$ at all. Hence, in order to fully specify the 3-D Pareto front in this example, we must use one extra nonlinear function $p_2 = f_2(p_1)$ to describe the tradeoff between p_1 and p_2. Furthermore, we also need to specify the range for the performance metric $p_1 \in [p_{1,\text{MIN}}, p_{1,\text{MAX}}]$ where $p_{1,\text{MIN}}$ and $p_{1,\text{MAX}}$ stand for the minimum and maximum possible values for p_1, respectively. With these three components: $p_3 = f_3(p_1, p_2)$, $p_2 = f_2(p_1)$, and $p_1 \in [p_{1,\text{MIN}}, p_{1,\text{MAX}}]$, the 3-D Pareto front is now fully specified.

The aforementioned discussion for 3-D Pareto front can be extended to other high-dimensional cases. In general, an M-D Pareto front should be specified by the following $M-1$ nonlinear functions:

$$\begin{aligned} p_M &= f_M(\mathbf{p}_M) = f_M(p_1, p_2, \ldots, p_{M-1}) \\ p_{M-1} &= f_{M-1}(\mathbf{p}_{M-1}) = f_{M-1}(p_1, p_2, \ldots, p_{M-2}) \\ &\vdots \\ p_2 &= f_2(\mathbf{p}_2) = f_2(p_1) \end{aligned} \tag{3}$$

and a 1-D interval representing the range of the performance metric p_1

$$p_1 \in \left[p_{1,\mathrm{MIN}}, p_{1,\mathrm{MAX}}\right] \tag{4}$$

where $\mathbf{p}_m = [p_1 p_2 \cdots p_{m-1}]^T \in \Re^{m-1}$ for any $m \in \{2, 3, \ldots, M\}$. The functions $\{f_m(\mathbf{p}_m); m = 2, 3, \ldots, M\}$ must be monotonically decreasing so that any point on the Pareto front satisfies the condition in (1) and, hence, is Pareto optimal.

To determine a Pareto front, we must find the closed-form expressions for the monotonic functions in (3). To this end, we adopt the linear regression technique from the statistics community [21]. Namely, we approximate each nonlinear function by the linear combination of a set of basis functions

$$p_m = f_m(\mathbf{p}_m) \approx g_m(\mathbf{p}_m) = \sum_{n=1}^{N_m} \alpha_{m,n} \cdot b_{m,n}(\mathbf{p}_m) \tag{5}$$

where $\{\alpha_{m,n}; m = 2, 3, \ldots, M, n = 1, 2, \ldots, N_m\}$ denote the model coefficients, $\{b_{m,n}(\mathbf{p}_m); m = 2, 3, \ldots, M, n = 1, 2, \ldots, N_m\}$ stand for the basis functions, and N_m is the total number of basis functions to approximate $f_m(\mathbf{p}_m)$. Note that the nonlinear functions $\{f_m(\mathbf{p}_m); m = 2, 3, \ldots, M\}$ have different input variables and dimensionalities, as shown in (3).

To solve the model coefficients $\{\alpha_{m,n}; n = 1, 2, \ldots, N_m\}$ associated with the Pareto front $f_m(\mathbf{p}_m)$, we first collect a set of Pareto optimal points $\{\mathbf{p}_m^{(k)}; k = 1, 2, \ldots, K_m\}$ where K_m is the total number of these points. Next, the model coefficients $\{\alpha_{m,n}; n = 1, 2, \ldots, N_m\}$ can be solved by the following constrained optimization:

$$\begin{aligned} \underset{\alpha_{m,1},\ldots,\alpha_{m,Nm}}{\text{minimize}} \quad & \frac{1}{2} \cdot \sum_{k=1}^{K_m} \left[g_m^{(k)} - f_m^{(k)}\right]^2 \\ \text{subject to} \quad & g_m^{(k)} \geq f_m^{(k)} \quad (k = 1, 2, \ldots, K_m) \\ & \frac{\partial g_m}{\partial p_i} \leq 0 \quad (i = 1, 2, \ldots, m-1) \end{aligned} \tag{6}$$

where $\boldsymbol{\alpha}_m = [\alpha_{m,1}\alpha_{m,2} \cdots \alpha_{m,Nm}]^T$, and $f_m^{(k)}$ and $g_m^{(k)}$ denote the values of the actual Pareto front $f_m(\mathbf{p}_m)$ and the approximated Pareto front $g_m(\mathbf{p}_m)$ at the kth Pareto optimal point $\mathbf{p}_m^{(k)}$ respectively.

The optimization formulation in (6) has a threefold meaning. First, the difference between the Pareto optimal points and the approximated Pareto front is quantitatively measured by the cost function that should be minimized. Second, a Pareto front defines the boundary of the feasible region. Hence, there exists no feasible point which can sit above the Pareto front, as guaranteed by the boundary constraints $\{g_m^{(k)} \geq f_m^{(k)}; k = 1, 2, \ldots, K_m\}$ in (6). Third, since the approximated Pareto front should be monotonically decreasing, the first-order derivative of $g_m(\mathbf{p}_m)$ with respect to each performance metric p_i should be nonpositive, as defined by the derivative constraints $\{\partial g_m(\mathbf{p}_m)/\partial p_i \leq 0; i = 1, 2, \ldots, m-1\}$ in (6).

Note that the number of optimization variables [i.e., $\boldsymbol{\alpha}_m = (\alpha_{m,1}\alpha_{m,2} \cdots \alpha_{m,Nm})^T$] in (6) is finite (i.e., N_m). However, the derivative constraints are defined over an infinite set (i.e., the entire feasible region). Conceptually, since the derivative constraints must hold at any feasible point inside the feasible region, there are an infinite number of constraints associated with the optimization formulation in (6). Such an optimization problem is referred to as semi-infinite programming in [23].

It is important to emphasize that a number of technical details must be carefully considered in order to solve the model coefficients $\boldsymbol{\alpha}_m = [\alpha_{m,1}\alpha_{m,2} \cdots \alpha_{m,Nm}]^T$ from the optimization problem in (6).

1) *Pareto Optimal Point Selection:* In our application, since the performance data are collected from online resources, many data points may not be Pareto optimal. In addition, as shown in Fig. 2, even if a feasible point sits on the 3-D Pareto front, it may or may not sit on the 2-D Pareto front. For the example in Fig. 2, there are five Pareto optimal points (denoted as the black dots) for the 3-D Pareto front, but only three of them (denoted as the red stars) are Pareto optimal for the 2-D Pareto front after projection. It, in turn, implies that we cannot use the same set of Pareto optimal points in (6) to fit all nonlinear functions $\{f_m(\mathbf{p}_m); m = 2, 3, \ldots, M\}$. For these reasons, it is extremely important for us to develop a data preprocessing technique that can automatically determine the appropriate set of Pareto optimal points to fit a particular nonlinear function $f_m(\mathbf{p}_m)$.
2) *Basis Function Selection:* Since the shape of the Pareto front is unknown, we do not know the appropriate basis functions in advance. If a set of wrong basis functions are used to approximate $f_m(\mathbf{p}_m)$, the estimated Pareto front $g_m(\mathbf{p}_m)$ may not be accurate. For this reason, a "smart" algorithm must be developed to automatically choose the appropriate basis functions for a given Pareto front.
3) *Semi-Infinite Programming:* Since the estimated Pareto front $g_m(\mathbf{p}_m)$ should be monotonically decreasing, the semi-infinite programming in (6) should verify that the optimal solution satisfies an infinite number of derivative constraints over the entire feasible region. Toward this goal, an efficient numerical algorithm must be adopted to deal with these derivative constraints.

In what follows, we will describe a number of numerical techniques to address these technical challenges and, consequently, make the proposed Pareto front modeling applicable to practical problems.

IV. Implementation Details

Our proposed high-dimensional Pareto front modeling is made of practical utility by applying a number of efficient numerical algorithms. In this section, we discuss these algorithms in detail and highlight their novelty.

A. Data Collection

As mentioned in Section II, layout parasitics and manufacturing nonidealities are often ignored by the conventional

Pareto front modeling methods [5]–[17]. Consequently, the Pareto fronts calculated by these conventional approaches may not accurately capture the postsilicon reality. In practice, it is almost impossible to consider all postsilicon nonidealities by presilicon simulation. Hence, instead of fully relying on presilicon simulation, there is a strong need to explore other alternative approaches to collect silicon-proved performance data for Pareto front modeling.

In this paper, we exploit the fact that a large amount of silicon measurement data can be found on the Internet, including technical papers published at various conferences and journals, data sheets provided by commercial companies, etc. Even though these performance data may not be exactly Pareto optimal in theory, they represent the "best" performance metrics that have been achieved in the literature. In addition, since the circuit performances reported from these sources are validated in silicon, they are expected to be more accurate than the presilicon simulation results. Furthermore, large-scale analog circuits or even systems have been measured in silicon and, hence, their performance data are also publically available from these online sources.

The aforementioned online data can be automatically collected and processed by using text mining techniques [19]. Text mining attempts to derive high-quality information from different text resources by analyzing the patterns and trends based on statistical learning [25]. It involves information retrieval, machine learning, computational linguistics, etc. [26]. As a well-established domain in the machine learning community, a large number of off-the-shelf text mining tools are commercially available on the market (e.g., statistical analysis system (SAS) [27] and IBM statistical product and service solutions (SPSS) [28]) and they have been successfully applied to a variety of practical applications (e.g., bioinformatics, national security, and sentiment analysis). We can directly adopt these text mining tools to collect analog performance data from the Internet for our application of Pareto front modeling.

It is important to note that our proposed data harvesting from the Internet are not applicable to the most advanced manufacturing technologies. Practically speaking, once a new manufacturing technology is available, it often takes one or two years to design the analog circuits and then collect the silicon measurement data. However, even with this limitation, the proposed approach can still be applied to a broad range of applications (e.g., biomedical circuits [29], [30], automotive electronics [31], [32], and Internet of things [33], [34]) where mature manufacturing technologies are preferred due to their high reliability and/or low cost.

B. Data Preprocessing

As mentioned at the end of Section III, many data points collected from online resources may not be Pareto optimal. Hence, we must carefully select the Pareto optimal points and only use them to fit the Pareto front of interest. Note that such a Pareto optimal point selection must be repeatedly performed for each nonlinear function $f_m(\mathbf{p}_m)$ where $m \in \{2, 3, \ldots, M\}$, because the Pareto optimal points are different for different functions, as shown by the simple example in Fig. 2.

Toward this goal, we formulate an efficient data preprocessing algorithm to quickly find all Pareto optimal points from a given data set (say, Θ) that contains a number of feasible points. As defined in Section III, a point $\mathbf{p}$ is Pareto optimal, if and only if it is not dominated by any other feasible point. Based on this definition, we first identify one Pareto optimal point (say, $\mathbf{p}_A$) that is not dominated by any other feasible point in the data set Θ. This step can be easily done by choosing the feasible point that has the greatest value for one of the performance metrics (say, p_1). We then remove all feasible points in Θ that are dominated by $\mathbf{p}_A$. Next, we select the feasible point (say, $\mathbf{p}_B$) from Θ that has the second greatest value for the performance metric p_1. We know that $\mathbf{p}_B$ is again a Pareto optimal point because of the following two reasons. First, $\mathbf{p}_B$ is not dominated by $\mathbf{p}_A$; otherwise, $\mathbf{p}_B$ should have already been removed from Θ. Second, $\mathbf{p}_B$ is not dominated by any other feasible point in Θ, since its value for the performance metric p_1 is greater than any other feasible point except $\mathbf{p}_A$. The aforementioned selection process is repeatedly applied until all Pareto optimal points are found.

Algorithm 1 Pareto Optimal Point Selection

1. Start from a given data set Θ containing a number of feasible points.
2. Initialize the set $\Xi = \{\}$.
3. Find the Pareto optimal point $\mathbf{p}$ from the data set Θ that has the greatest value for the first performance metric p_1.
4. Remove all feasible points in Θ that are dominated by $\mathbf{p}$.
5. Remove $\mathbf{p}$ from the set Θ, and add $\mathbf{p}$ to the set Ξ.
6. If the set Θ is not empty, go to Step 3. Otherwise, stop iteration and the set Ξ contains all Pareto optimal points.

Algorithm 1 summarizes the simplified flow of our proposed data preprocessing steps. Note that Algorithm 1 does not consider the special case where there exist multiple feasible points having the same greatest value for the performance metric p_1. However, it is straightforward to extend Algorithm 1 to handle such a special case and, hence, more details are not included in this paper.

C. Sparse Regression

In addition to Pareto optimal point selection, basis function selection is another critical issue that must be carefully addressed. As previously discussed, we do not know the optimal basis functions in advance: they strongly depend on the Pareto front to be modeled, and may vary from technology to technology, from circuit to circuit, and from performance to performance. Hence, it is impossible to come up with a fixed set of basis functions that are applicable to all cases. Instead, the appropriate basis functions must be adaptively selected for a particular Pareto front of interest on the fly.

In this paper, sparse regression [20]–[22] could be adopted for automatic basis function selection. In particular, our implementation of sparse regression is based on the idea of stepwise basis function selection [21]. It applies a greedy algorithm to iteratively select a subset of important basis functions from a large pool of candidates by minimizing the modeling error. In what follows, we describe the sparse regression algorithm in detail.

We start from a large number of possible candidates of basis functions $\{b_{m,n}(\mathbf{p}_m); n = 1, 2, \ldots, N_m\}$ (e.g., trigonometric

Algorithm 2 Sparse Regression

1. Start from a set of possible candidates of basis functions $\{b_{m,n}(\mathbf{p}_m); n = 1, 2, \ldots, N_m\}$, a number of Pareto optimal points $\{\mathbf{p}_m^{(k)}; k = 1, 2, \ldots, K_m\}$, and a constant λ_m representing the total number of basis functions that should be selected to approximate the nonlinear function $f_m(\mathbf{p}_m)$ in (5).
2. Initialize the index sets $\Omega = \{1, 2, \ldots, N_m\}$ and $\Psi = \{\}$.
3. For each $n \in \Omega$, combine $b_{m,n}(\mathbf{p}_m)$ with all other basis functions $\{b_{m,l}(\mathbf{p}_m); l \in \Psi\}$ to approximate $f_m(\mathbf{p}_m)$ where the model coefficients are solved by the following constrained optimization:

$$\begin{aligned} &\underset{a_{m,l}, l\in\{n\}\cup\Psi}{\text{minimize}} && \frac{1}{2}\cdot\sum_{k=1}^{K_m}\left[\sum_{l\in\{n\}\cup\Psi}\alpha_{m,l}\cdot b_{m,l}\left(\mathbf{p}_m^{(k)}\right)-f_m^{(k)}\right]^2 \\ &\text{subject to} && \sum_{l\in\{n\}\cup\Psi}\alpha_{m,l}\cdot b_{m,l}\left(\mathbf{p}_m^{(k)}\right)\geq f_m^{(k)} \\ &&& (k=1,2,\ldots,K_m). \\ &&& \sum_{l\in\{n\}\cup\Psi}\alpha_{m,l}\cdot\frac{\partial b_{m,l}(\mathbf{p}_m)}{\partial p_i}\leq 0 \\ &&& \begin{pmatrix} i=1,2,\ldots,m-1 \\ \forall\mathbf{p}_m \end{pmatrix} \end{aligned} \tag{7}$$

4. Based on the model coefficients $\{\alpha_{m,l}; l \in \{n\} \cup \Psi\}$ solved from (7), calculate the modeling error for each $n \in \Omega$. Find the optimal index (say, n_{OPTI}) for which the modeling error is minimal.
5. Remove n_{OPTI} from the set Ω, and add n_{OPTI} to the set Ψ.
6. If the cardinality of the set Ψ (i.e., the number of elements in the set Ψ) is less than λ_m, go to Step 3. Otherwise, stop iteration and the set Ψ contains the indexes for the selected basis functions.

basis functions) to approximate the nonlinear function $f_m(\mathbf{p}_m)$ in (5). Initially, without knowing which basis function is important, we attempt to use each basis function from the set $\{b_{m,n}(\mathbf{p}_m); n = 1, 2, \ldots, N_m\}$ to approximate $f_m(p_m)$. The optimal basis function [say, $b_{m,A}(\mathbf{p}_m)$] resulting in the minimum modeling error is chosen as an important basis function.

Next, given $b_{m,A}(\mathbf{p}_m)$ as a basis function that is already selected, we further combine $b_{m,A}(\mathbf{p}_m)$ with each of the other basis functions from the set $\{b_{m,n}(\mathbf{p}_m); n = 1, 2, \ldots, N_m, \neq A\}$ to approximate $f_m(\mathbf{p}_m)$. Here, we are looking for the optimal basis function [say, $b_{m,B}(\mathbf{p}_m)$] that results in the minimum modeling error, when $b_{m,B}(\mathbf{p}_m)$ is combined with $b_{m,A}(\mathbf{p}_m)$ to approximate $f_m(\mathbf{p}_m)$. Once $b_{m,B}(\mathbf{p}_m)$ is found, it is chosen as the second important basis function.

The aforementioned selection process is repeatedly applied until a sufficient number of basis functions are selected. At each iteration step, one basis function is chosen to minimize the modeling error. Algorithm 2 summarizes the major steps of the sparse regression algorithm.

There are several important clarifications that should be made for the sparse regression technique summarized by Algorithm 2. First, the stepwise approach taken by Algorithm 2 is heuristic and it does not guarantee global optimum. However, such a heuristic approach results in a locally optimal solution that is reasonably good for many practical applications [21]. As will be demonstrated by our numerical examples in Section V, Algorithm 2 facilitates us to successfully identify a small number of (e.g., $10^1 \sim 10^2$) important basis functions from a large number of (e.g., $10^2 \sim 10^3$) possible candidates to accurately model the Pareto font of interest.

Second, Algorithm 2 assumes that the number of required basis functions (i.e., λ_m) is given as the input. In practice, λ_m is unknown and must be automatically determined by a statistical technique that is referred to cross-validation in [21]. The key idea is to repeatedly run Algorithm 2 for different values of λ_m and calculate the modeling error associated with each λ_m. In order to accurately estimate the modeling error at each run, the set of Pareto optimal points is divided into a training set and a testing set that are not overlapped. The training set is used to solve the model coefficients, and the testing set is used to validate the model (i.e., estimate the modeling error). As such, the over-fitting problem can be easily detected. Once the modeling error is known, the optimal value of λ_m is determined by minimizing the modeling error. More details about cross-validation can be found in [21].

Third, solving the constrained optimization in (7) is not trivial, because the derivative constraints must hold over the entire feasible region (i.e., an infinite set). Equation (7) represents a semi-infinite programming problem and an efficient numerical algorithm must be adopted to solve it, as will be discussed in detail in the next section.

Finally, since Algorithm 2 solves the modeling problem by sparse regression, the candidates of basis functions must be carefully chosen. In theory, these basis functions should satisfy the restricted isometry property (RIP) so that the sparse model coefficients can be accurately solved [35]–[37]. Based on this requirement, trigonometric basis is one of the appropriate choices, while wavelet basis or radial basis is not applicable. More details about the RIP can be found in the literature from the statistics community [35]–[37].

In addition to sparse regression, there exist a number of other advanced regression modeling techniques in the literature (e.g., Gaussian process model and kernel ridge regression) [21], [25], [38]. However, most of these techniques cannot easily handle the boundary constraints and/or the derivative constraints and, hence, are not applicable to our application of Pareto front modeling. For these reasons, we do not discuss other regression modeling techniques in this paper.

D. Grid Discretization Method

A variety of numerical algorithms have been developed to solve semi-infinite programming problems in the mathematics community [23]. It has been shown that with several general assumptions, the optimal solution of a semi-infinite programming problem can be found by iteratively solving a sequence of finite programming problems with a finite number of constraints only.

In this paper, we will adopt the grid discretization method [23] to solve (7). Given the performance vector $\mathbf{p}_m \in \Re^{m-1}$, we first choose a set of initial step sizes

$\{h_i^{(0)}; i = 1, 2, \ldots, m-1\}$, where $h_i^{(0)}$ represents the step size to discretize the performance metric p_i (i.e., the ith element of $\mathbf{p}_m$). Given these step sizes, the feasible region of $\mathbf{p}_m$ is discretized to a grid $G_m^{(0)}$. Note that the size of $G_m^{(0)}$ exponentially increases with the dimensionality of the performance space. Hence, instead of keeping the entire grid $G_m^{(0)}$, we randomly select a set of points from $G_m^{(0)}$ to construct a reduced grid $G_{Rm}^{(0)}$ (i.e., $G_{Rm}^{(0)} \subseteq G_m^{(0)}$).

We solve the following optimization problem by forcing the derivative constraints on the grid $G_{Rm}^{(0)}$, instead of the entire feasible region:

$$\begin{aligned} \underset{a_{m,l}, l\in\{n\}\cup\Psi}{\text{minimize}} \quad & \frac{1}{2} \cdot \sum_{k=1}^{K_m} \left[\sum_{l\in\{n\}\cup\Psi} \alpha_{m,l} \cdot b_{m,l}\left(\mathbf{p}_m^{(k)}\right) - f_m^{(k)} \right]^2 \\ \text{subject to} \quad & \sum_{l\in\{n\}\cup\Psi} \alpha_{m,l} \cdot b_{m,l}\left(\mathbf{p}_m^{(k)}\right) \geq f_m^{(k)} \\ & (k = 1, 2, \ldots, K_m) \\ & \sum_{l\in\{n\}\cup\Psi} \alpha_{m,l} \cdot \frac{\partial b_{m,l}(\mathbf{p}_m)}{\partial p_i} \leq 0 \\ & \begin{pmatrix} i = 1, 2, \ldots, m-1 \\ \mathbf{p}_m \in G_{Rm}^{(0)} \end{pmatrix}. \end{aligned} \tag{8}$$

Since the set $G_{Rm}^{(0)}$ is finite, (8) is no longer a semi-infinite programming problem. In addition, a close examination of (8) reveals two important observations. First, the cost function is quadratic and convex with respect to the problem unknowns $\{\alpha_{m,l}; l \in \{n\} \cup \Psi\}$. Second, all constraints are linear functions of $\{\alpha_{m,l}; l \in \{n\} \cup \Psi\}$. For these reasons, (8) is a convex optimization problem that can be solved both efficiently (i.e., with low computational cost) and robustly (i.e., with guaranteed global optimum) [39].

Once the model coefficients $\{\alpha_{m,l}; l \in \{n\} \cup \Psi\}$ are found, we check the derivative constraints on the entire grid $G_m^{(0)}$. If the constraints are not fully satisfied, a number of new points from $G_m^{(0)}$ should be added to $G_{Rm}^{(0)}$, and then we solve the optimization in (8) to update the model coefficients $\{\alpha_{m,l}; l \in \{n\} \cup \Psi\}$. In other words, we will keep increasing the number of points in $G_{Rm}^{(0)}$ and solving the model coefficients until the derivative constraints hold for the entire grid $G_m^{(0)}$.

The initial step sizes $\{h_i^{(0)}; i = 1, 2, \ldots, m-1\}$ are often set to a series of large values, thereby resulting in a coarse grid $G_m^{(0)}$. Even though the Pareto front solved from (8) satisfies the derivative constraints on the grid $G_m^{(0)}$, it may not be monotonically decreasing over the entire feasible region. Therefore, we must further refine the grid and enforce the derivative constraints. To this end, we set

$$h_i^{(1)} = \frac{h_i^{(0)}}{s} \quad (i = 1, 2, \ldots, m-1) \tag{9}$$

where $s \geq 2$ is a positive integer. Given the updated step sizes $\{h_i^{(1)}; i = 1, 2, \ldots, m-1\}$, we first generate a new grid $G_m^{(1)}$, and then select a set of points from $G_m^{(1)}$ to construct the reduced grid $G_{Rm}^{(1)}$ where $G_{Rm}^{(0)} \subseteq G_{Rm}^{(1)} \subseteq G_m^{(1)}$.

Replacing $G_{Rm}^{(0)}$ by $G_{Rm}^{(1)}$ in (8), we solve the convex optimization again and update the model coefficients $\{\alpha_{m,l}; l \in \{n\} \cup \Psi\}$. Similar to the case of $G_{Rm}^{(0)}$, we will keep increasing the size of $G_{Rm}^{(1)}$ and solving the model coefficients until the derivative constraints hold for the entire grid $G_m^{(1)}$. Next, we further reduce the step sizes and repeat the aforementioned procedure until the step sizes are sufficiently small (e.g., less than a predefined threshold). Algorithm 3 summarizes the major steps of the grid discretization method to solve the semi-infinite programming problem in (7).

We can further apply a number of heuristics to improve the computational efficiency of Algorithm 3. First, we observe that the model coefficients $\{\alpha_{m,l}; l \in \{n\} \cup \Psi\}$ solved from (11) are often similar between two successive iterations. Hence, when solving these model coefficients for the qth iteration, we can take the solution from the $(q-1)$th iteration as the initial guess. Second, when constructing the reduced grid $G_{Rm}^{(q)}$, we should first include all points from $G_{Rm}^{(q-1)}$ such that $G_{Rm}^{(q-1)} \subseteq G_{Rm}^{(q)}$. Next, when adding additional points from $G_m^{(q)}$ to $G_{Rm}^{(q)}$, we choose the points where the derivative constraints are greatly violated according to the model coefficients $\{\alpha_{m,l}; l \in \{n\} \cup \Psi\}$ solved

---

Algorithm 3 Grid Discretization Method

1. Start from a set of initial step sizes $\{h_i^{(0)}; i = 1, 2, \ldots, m-1\}$.
2. Construct the initial grid $G_m^{(0)}$ and the reduced grid $G_{Rm}^{(0)}$. Solve the optimization problem in (8) to determine the model coefficients $\{\alpha_{m,l}; l \in \{n\} \cup \Psi\}$.
3. If the derivative constraints do not hold for the entire grid $G_m^{(0)}$, add a number of new points from $G_m^{(0)}$ to $G_{Rm}^{(0)}$, and go back to Step 2. Otherwise, go to Step 4.
4. Set the iteration index $q = 1$.
5. Reduce the step sizes:
$$h_i^{(q)} = \frac{h_i^{(q-1)}}{s} \quad (i = 1, 2, \ldots, m-1). \tag{10}$$
6. Construct the grid $G_m^{(q)}$ based on (10) and construct the reduced grid $G_{Rm}^{(q)}$ such that $G_{Rm}^{(q-1)} \subseteq G_{Rm}^{(q)} \subseteq G_m^{(q)}$.
7. Solve the model coefficients $\{\alpha_{m,l}; l \in \{n\} \cup \Psi\}$ from the following optimization:
$$\begin{aligned} \underset{a_{m,l}, l\in\{n\}\cup\Psi}{\textit{minimize}} \quad & \frac{1}{2} \cdot \sum_{k=1}^{K_m} \left[\sum_{l\in\{n\}\cup\Psi} \alpha_{m,l} \cdot b_{m,l}\left(\mathbf{p}_m^{(k)}\right) - f_m^{(k)} \right]^2 \\ \textit{subject to} \quad & \sum_{l\in\{n\}\cup\Psi} \alpha_{m,l} \cdot b_{m,l}\left(\mathbf{p}_m^{(k)}\right) \geq f_m^{(k)} \\ & (k = 1, 2, \ldots, K_m). \\ & \sum_{l\in\{n\}\cup\Psi} \alpha_{m,l} \cdot \frac{\partial b_{m,l}(\mathbf{p}_m)}{\partial p_i} \leq 0 \\ & \begin{pmatrix} i = 1, 2, \ldots, m-1 \\ \mathbf{p}_m \in G_{Rm}^{(q)} \end{pmatrix} \end{aligned} \tag{11}$$
8. If the derivative constraints do not hold for the entire grid $G_m^{(q)}$, add a number of new points from $G_m^{(q)}$ to $G_m^{(q)}$, and go back to Step 7. Otherwise, go to Step 9.
9. If the step sizes $\{h_i^{(q)}; i = 1, 2, \ldots, m-1\}$ are sufficiently small, stop iteration. Otherwise, set $q = q + 1$ and go to Step 5.

Algorithm 4 Pareto Front Modeling

1. Start from a set of feasible points $\mathbf{p} \in \Re^M$ (i.e., the feasible performance metrics) that are collected from online resources.
2. For each performance metric, p_m, where $m \in \{1, 2, \ldots, M\}$, determines its lower bound $p_{m,MIN}$ and upper bound $p_{m,MAX}$.
3. For $m = 2, 3, \ldots, M$
4. Apply Algorithm 1 to select the Pareto optimal points $\{\mathbf{p}_m^{(k)}; k = 1, 2, \ldots, K_m\}$ from the feasible points.
5. Determine the optimal number of required basis functions (i.e., λ_m) for the nonlinear function $f_m(\mathbf{p}_m)$ in (5) by running Algorithms 2 and 3 with cross-validation.
6. Given the optimal value of λ_m, apply Algorithms 2 and 3 to find the optimal basis functions $\{b_{m,n}(\mathbf{p}_m); n \in \Psi\}$ and the corresponding model coefficients $\{\alpha_{m,n}; n \in \Psi\}$ based on all Pareto optimal points$\{\mathbf{p}_m^{(k)}; k = 1, 2, \ldots, K_m\}$.
7. Use the basis functions $\{b_{m,n}(\mathbf{p}_m); n \in \Psi\}$ and the model coefficients $\{\alpha_{m,n}; n \in \Psi\}$ to construct the nonlinear function $g_m(\mathbf{p}_m)$ in (5).
8. End For
9. The Pareto front of interest is defined by the $M-1$ nonlinear functions in (3) and the 1-D interval in (4).

from the $(q-1)$th iteration. In practice, we find that the aforementioned heuristics can often reduce the total number of iterations and, hence, the overall computational cost of Algorithm 3.

E. Summary

Algorithm 4 summarizes the overall flow of our proposed high-dimensional Pareto front modeling approach by using online resources. Starting from a set of feasible points, Algorithm 4 first identifies the Pareto optimal points from the given data set and then applies sparse regression with grid discretization to fit the high-dimensional Pareto front of interest.

Compared to the existing methods based on presilicon transistor-level simulations, Algorithm 4 adopts a completely different strategy for Pareto front modeling. It takes full advantage of the design knowledge harvested from the Internet (i.e., the postsilicon measurement data for analog circuits) to learn Pareto fronts. Consequently, layout parasitics and manufacturing nonidealities can be accurately captured while expensive transistor-level simulations are no longer required. Furthermore, our proposed method produces a compact expression for high-dimensional Pareto front by formulating a sparse regression problem with consideration of both boundary and derivative constraints. An efficient algorithm of semi-infinite programming is adopted to robustly solve the sparse model coefficients based on grid discretization. While the concepts of sparse regression and semi-infinite programming are borrowed from the literature, the novelty of our proposed Pareto front modeling lies in the unique integration and tuning of these existing methodologies (e.g., by adding boundary and derivative constraints) for our specific application of interest.

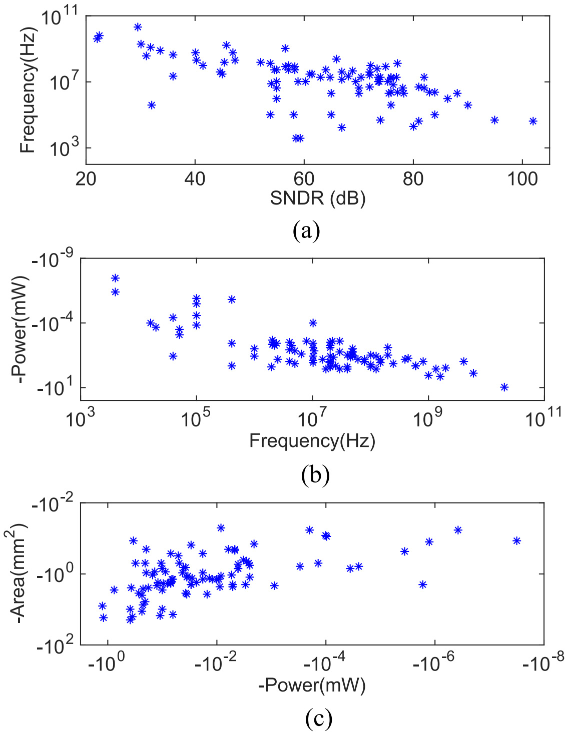


Fig. 3. ADC performance data are collected for the 0.18 μm technology node [41]. (a) SNDR (p_1) versus frequency (p_2). (b) Frequency (p_2) versus -power (p_3). (c) -Power (p_3) versus -area (p_4).

V. Numerical Experiments

In this section, we demonstrate the efficacy of our proposed algorithm for Pareto front modeling by two circuit examples. In our experiments, trigonometric functions (i.e., sine and cosine functions) are used to define the basis functions in (5), where ten basis functions are set for each dimension. Namely, to approximate an M-D Pareto front, 10^{M-1} basis functions in total are considered as the possible candidates for basis selection. All numerical experiments are performed on a workstation with 2 GHz CPU and 25 GB memory.

A. Analog-to-Digital Converter

In this section, we model the Pareto fronts of ADC for four different performance metrics: 1) SNDR (p_1); 2) Nyquist sampling frequency (p_2); 3) -power (p_3); and 4) -area (p_4). Note that in order to follow the definition of Pareto optimal point in (1), we use p_3 to denote -power (instead of + power) and p_4 to denote -area (instead of + area). Such choices are made, because small power and area are preferred for an ADC design [40].

We use the ADC performance data [41] at the 0.18 μm technology node to construct three nonlinear functions: $p_2 = f_2(p_1)$, $p_3 = f_3(p_1, p_2)$, and $p_4 = f_4(p_1, p_2, p_3)$, as shown in (3). Since these data points are 4-D and cannot be directly visualized due to their high dimensionality, we show them by using three 2-D plots in Fig. 3. Note that strong correlations can be observed for the data points in these 2-D plots. For instance, SNDR increases as frequency decreases, as shown in Fig. 3(a). It, in turn, indicates the fundamental tradeoffs between these performance metrics.

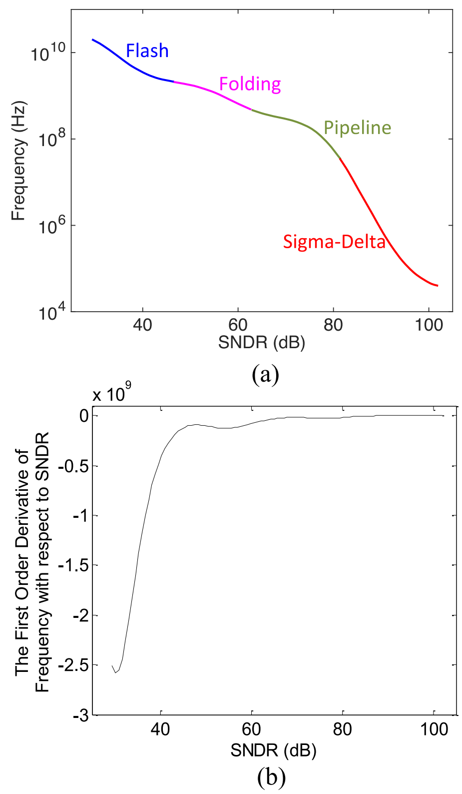


Fig. 4. Results of Pareto front modeling are shown for SNDR (p_1) and frequency (p_2). (a) Tradeoff between SNDR and frequency described by the nonlinear function $p_2 = f_2(p_1)$ where different ADC architectures are labeled by different colors. (b) First-order derivative $\mathrm{d}p_2/\mathrm{d}p_1$ of frequency with respect to SNDR.

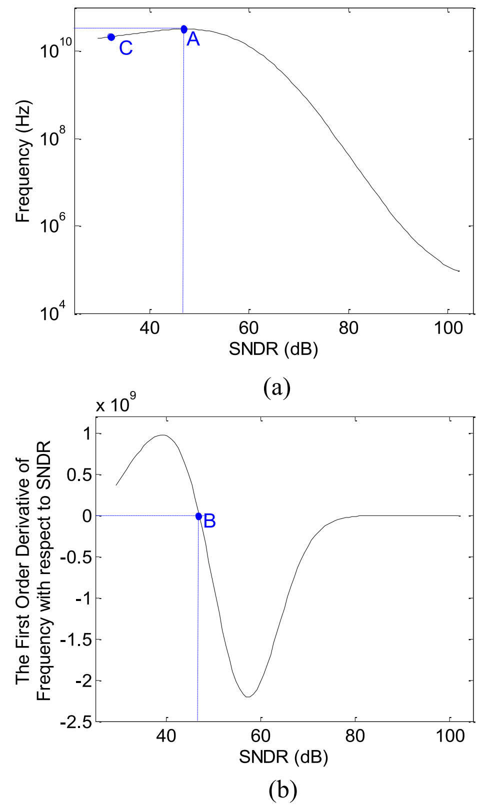


Fig. 5. Results of Pareto front modeling without enforcing the derivative constraints are shown for SNDR (p_1) and frequency (p_2). (a) Tradeoff between SNDR and frequency described by the nonlinear function $p_2 = f_2(p_1)$ where the point "A" represents the largest frequency value. (b) First-order derivative $\mathrm{d}p_2/\mathrm{d}p_1$ of frequency with respect to SNDR where the derivative reaches zero at the point "B."

The performance data in [41] cover a variety of ADC architectures (e.g., flash ADC, pipeline ADC, folding ADC, and Sigma–Delta ADC) and are reported by the papers published at the ISSCC or the VLSI Symposium. In this example, even though we directly borrow the ADC performance data from [41], there is nothing that prevents us from running a text mining tool (e.g., SAS [27]) to directly harvest the data from the Internet.

To the best of our knowledge, most existing algorithms for Pareto front modeling can only capture the Pareto fronts for small analog circuits [5]–[17]. The 4-D Pareto front modeling problem attacked in this example is for a large-scale circuit (i.e., ADC) and has been considered to be extremely challenging in the literature.

Fig. 4 shows the modeling results for the 2-D Pareto front $p_2 = f_2(p_1)$ that captures the tradeoff between SNDR and frequency. Note that the approximated 2-D Pareto front is smooth and monotonically decreasing. Its first-order derivative $\mathrm{d}p_2/\mathrm{d}p_1$ is nonpositive, as specified by the given derivative constraints. Fig. 4(a) also shows different ADC architectures for different SNDR and frequency values, as reported by [41]. Note that the flash architecture is preferred at high frequency while the Sigma–Delta architecture is preferred at low frequency. In other words, the Pareto front in Fig. 4(a) facilitates us to choose the appropriate ADC architecture based on the required frequency and/or SNDR.

For testing and comparison purposes, we remove the derivative constraints from our optimization formulation in (6) and build a different 2-D Pareto front model $p_2 = f_2(p_1)$ as shown in Fig. 5. Studying Fig. 5 reveals an important observation that the nonlinear function $p_2 = f_2(p_1)$ is no longer monotonic, since the derivative constraints are not enforced. The function $p_2 = f_2(p_1)$ reaches its maximum at the point "A" in Fig. 5(a), where the corresponding first-order derivative $\mathrm{d}p_2/\mathrm{d}p_1$ is equal to zero as shown by the point "B" in Fig. 5(b). As a result, a number of points on the curve $p_2 = f_2(p_1)$ [e.g., the point "C" in Fig. 5(a)] are not Pareto optimal, as they are dominated by the point "A." Hence, the function $p_2 = f_2(p_1)$ in Fig. 5(a) is not a valid Pareto front.

Fig. 6 shows the 3-D Pareto front $p_3 = f_3(p_1, p_2)$ that captures the tradeoff between -power, SNDR and frequency. For ADC circuits, it is well-known that the power consumption increases as the SNDR or frequency increases. Hence, the first-order derivatives $\mathrm{d}p_3/\mathrm{d}p_1$ and $\mathrm{d}p_3/\mathrm{d}p_2$ of -power with respect to SNDR and frequency should be nonpositive in this example. These derivative constraints are enforced in our optimization formulation to solve the model coefficients. As a result, the Pareto front in Fig. 6(a) is monotonically decreasing. The nonpositive derivatives $\mathrm{d}p_3/\mathrm{d}p_1$ and $\mathrm{d}p_3/\mathrm{d}p_2$ are further plotted in Fig. 6(b) and (c).

For the 4-D Pareto front $p_4 = f_4(p_1, p_2, p_3)$ that captures the tradeoff between -area, -power, SNDR and frequency, the resulting nonlinear function $f_4(\bullet)$ is 3-D and cannot be easily visualized due to its high dimensionality. For this reason,

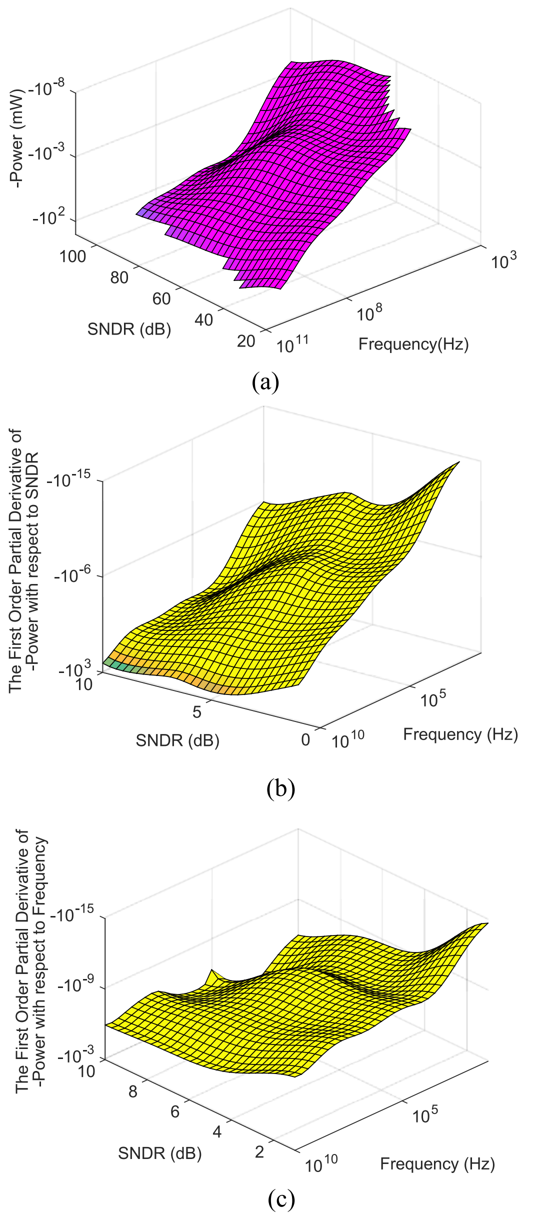


Fig. 6. Results of Pareto front modeling are shown for SNDR (p_1), frequency (p_2), and -power (p_3). (a) Tradeoff between -power, SNDR, and frequency described by the nonlinear function $p_3 = f_3(p_1, p_2)$. (b) First-order derivative $\partial p_3/\partial p_1$ of -power with respect to SNDR. (c) First-order derivative $\partial p_3/\partial p_2$ of -power with respect to frequency.

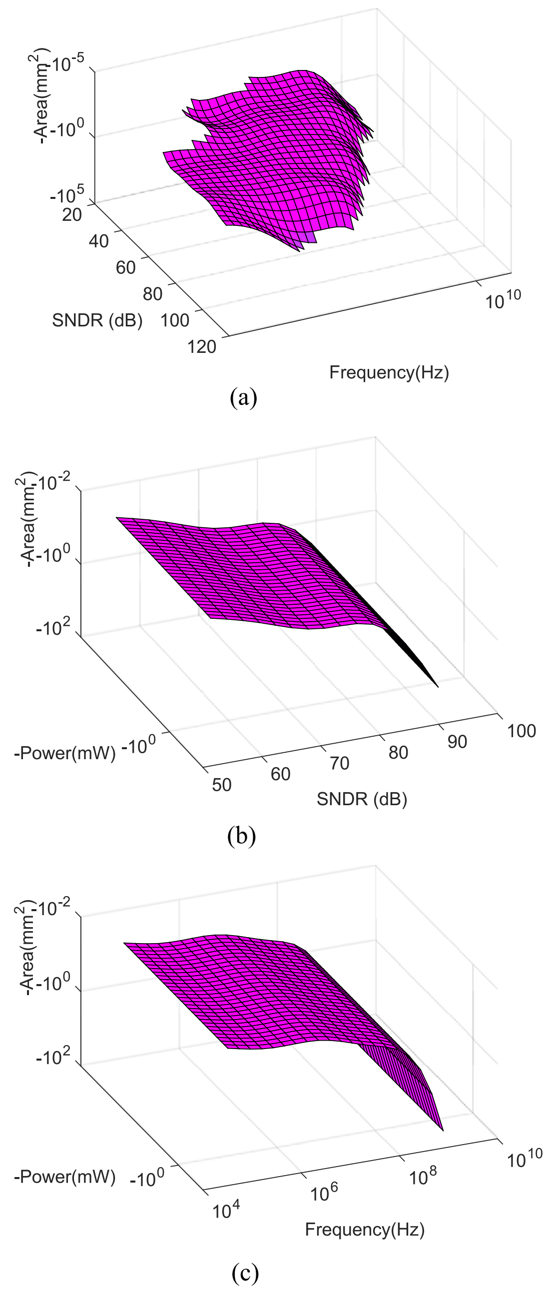


Fig. 7. Results of Pareto front modeling are shown for SNDR (p_1), frequency (p_2), -power (p_3), and -area (p_4). (a) Tradeoff between -area, SNDR, and frequency where power is fixed to 5.6×10^{-4} mW. (b) Tradeoff between -area, SNDR, and -power where frequency is fixed to 8.9 MHz. (c) Tradeoff between -area, frequency, and -power where SNDR is fixed to 62 dB.

TABLE I
PARETO FRONT MODELING RESULTS FOR ADC

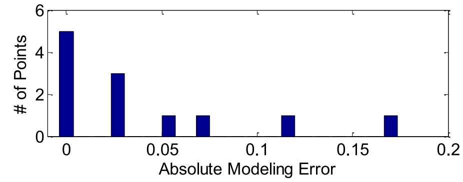
	$f_2(p_1)$	$f_3(p_1, p_2)$	$f_4(p_1, p_2, p_3)$
# of Pareto Optimal Points	12	47	61
# of Basis Functions	10	10^2	10^3
# of Selected Basis Functions	5	12	22
$Error_{RMS}$	0.066	0.08	0.1

we plot $f_4(\bullet)$ by using three 3-D plots as shown in Fig. 7. Due to the page limit, we do not plot the corresponding derivatives for $f_4(\bullet)$ here.

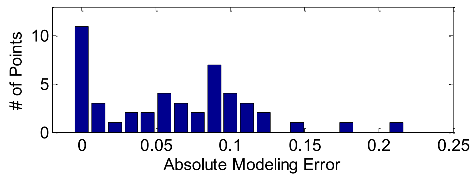
To quantitatively assess the modeling error, Fig. 8 shows the histograms of absolute modeling error at Pareto optimal points for three nonlinear functions: $p_2 = f_2(p_1)$, $p_3 = f_3(p_1, p_2)$ and $p_4 = f_4(p_1, p_2, p_3)$. Note that the modeling error is less than 0.1 for most points. In very few cases, the modeling error may reach 0.35; however, it is still valuable for many system-level design applications.

Table I further summarizes the modeling statistics for 2-D, 3-D, and 4-D Pareto fronts. In Table I, the root-mean-square (RMS) error $\text{Error}_{\text{RMS}}$ is defined as

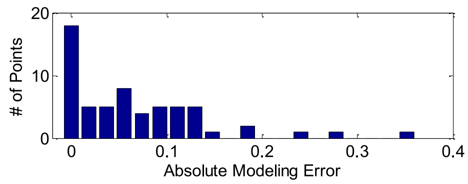
$$\text{Error}_{\text{RMS}} = \sqrt{\sum_{k=1}^{K_m} (y_k - \tilde{y}_k)^2 \Big/ K_m} \tag{12}$$



(a)

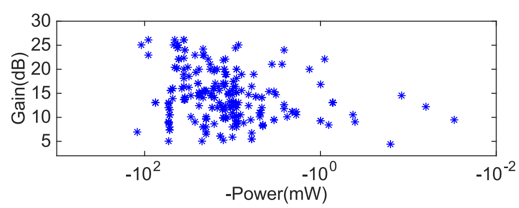


(b)

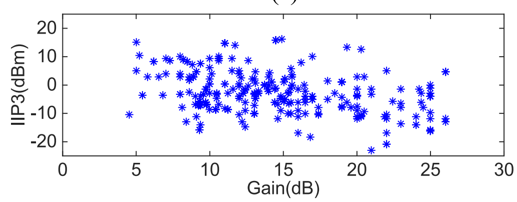


(c)

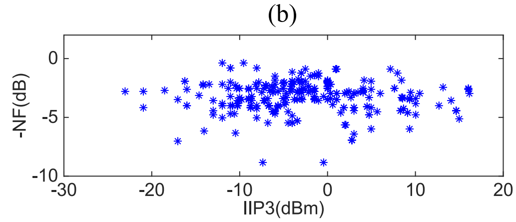
Fig. 8. Histograms of absolute modeling error at Pareto optimal points are shown for (a) $p_2 = f_2(p_1)$, (b) $p_3 = f_3(p_1, p_2)$, and (c) $p_4 = f_4(p_1, p_2, p_3)$. Each performance metric is normalized to the interval [0 1] in our experiments.



(a)



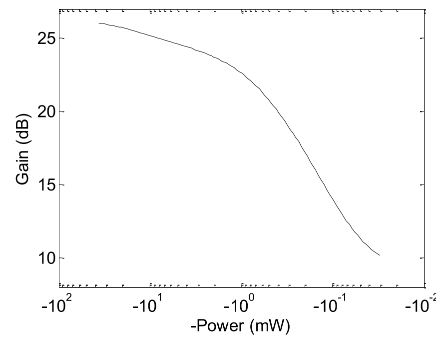
(b)



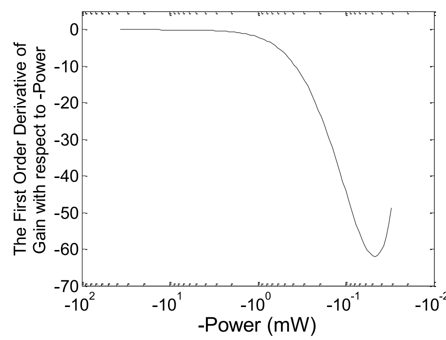
(c)

Fig. 9. LNA performance data are collected from the Internet. (a) -Power (p_1) versus gain (p_2). (b) Gain (p_2) versus IIP3 (p_3). (c) IIP3 (p_3) versus -NF (p_4).

where y_k and $\tilde{y}_k$ are the approximated and actual performance values after being normalized to the interval [0 1] at the kth Pareto optimal point respectively, and K_m is the total number of Pareto optimal points. Once the Pareto front models are built by applying Algorithm 4, all Pareto optimal points chosen by Algorithm 1 are used to evaluate the modeling error based on (12). However, it is important to mention that the Pareto optimal points used to approximate the nonlinear functions



(a)



(b)

Fig. 10. Results of Pareto front modeling are shown for -power (p_1) and gain (p_2). (a) Tradeoff between power and gain described by the nonlinear function $p_2 = f_2(p_1)$. (b) First-order derivative $\mathrm{d}p_2/\mathrm{d}p_1$ of gain with respect to -power.

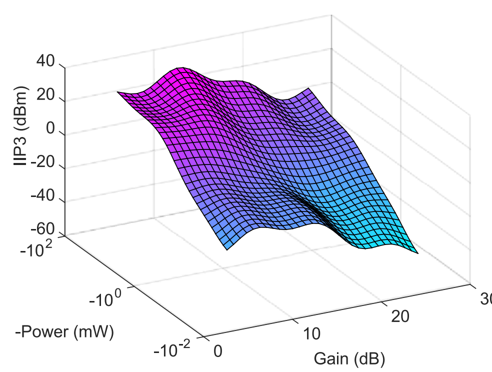
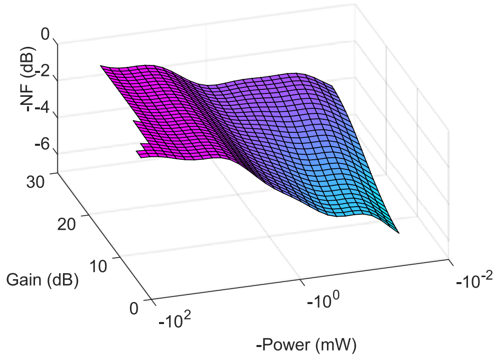


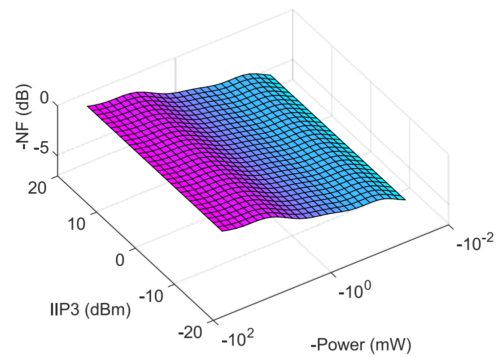
Fig. 11. Tradeoff between -power (p_1), gain (p_2), and IIP3 (p_3) is described by the nonlinear function $p_3 = f_3(p_1, p_2)$.

$p_2 = f_2(p_1)$, $p_3 = f_3(p_1, p_2)$ and $p_4 = f_4(p_1, p_2, p_3)$ are different, as explained in Fig. 2. Hence, the summation in (12) is performed over different data points when evaluating the errors for 2-D, 3-D, and 4-D Pareto fronts.

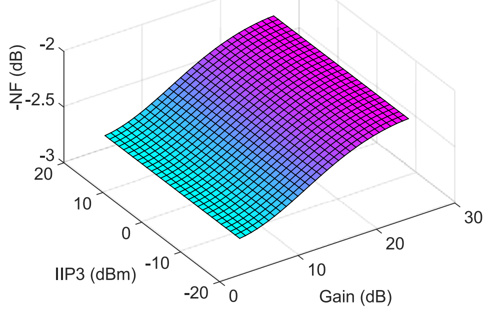
Studying Table I, we would have two important observations. First, even though there are a large number of basis functions, only a small number of Pareto optimal points are available for us to fit the Pareto fronts. In this case, the proposed sparse regression algorithm appropriately selects a small subset of important basis functions to approximate the Pareto fronts so that the results are not over-fitted due to the limited data set. Second, but more importantly, the RMS error $\mathrm{Error}_{\mathrm{RMS}}$ is less than 0.1 for all three Pareto front models: $p_2 = f_2(p_1)$, $p_3 = f_3(p_1, p_2)$, and $p_4 = f_4(p_1, p_2, p_3)$. It, in turn, demonstrates that our proposed algorithm accurately captures the Pareto fronts of interest in this example.



(a)



(b)



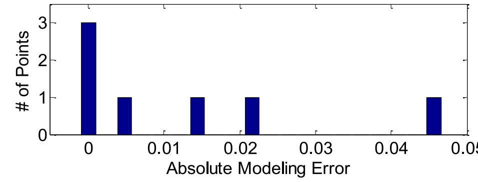
(c)

Fig. 12. Results of Pareto front modeling are shown for -power (p_1), gain (p_2), IIP3 (p_3), and -NF (p_4). (a) Tradeoff between -NF, -power, and gain where IIP3 is fixed to -1.2 dBm. (b) Tradeoff between -NF, -power, and -IIP3 where gain is fixed to 15.25 dB. (c) Tradeoff between -NF, gain, and IIP3 where power is fixed to 1.64 mW.

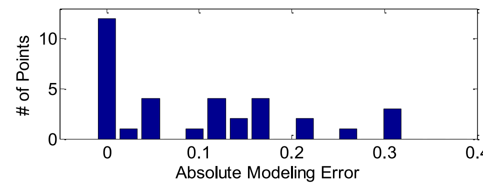
Compared to other existing approaches, the proposed Pareto front modeling method is based upon a completely different strategy to collect the performance data from the Internet. If a conventional Pareto front modeling technique is used, we need to run a large number of presilicon simulations to collect the required performance data. These simulations are not computationally affordable for large-scale ADC circuits even if we do not take into account layout parasitics and other manufacturing nonidealities. Hence, we do not show the results of the conventional Pareto front modeling methods for the ADC example here.

B. Low-Noise Amplifier

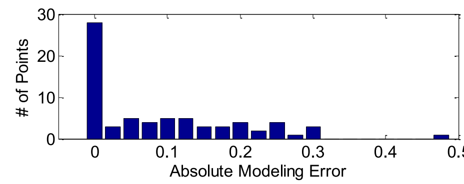
In this section, we aim to model the Pareto fronts of low-noise amplifier (LNA) with the consideration of four



(a)



(b)



(c)

Fig. 13. Histograms of absolute modeling error at Pareto optimal points are shown for (a) $p_2 = f_2(p_1)$, (b) $p_3 = f_3(p_1, p_2)$, and (c) $p_4 = f_4(p_1, p_2, p_3)$. Each performance metric is normalized to the interval [0 1] in our experiments.

TABLE II
PARETO FRONT MODELING RESULTS FOR LNA

	$f_2(p_1)$	$f_3(p_1, p_2)$	$f_4(p_1, p_2, p_3)$
# of Pareto Optimal Points	7	35	71
# of Basis Functions	10	10^2	10^3
# of Selected Basis Functions	4	19	22
$Error_{RMS}$	0.02	0.16	0.14

performance metrics: 1) -power (p_1); 2) gain (p_2); 3) IIP3 (p_3); and 4) -NF (p_4), where NF denotes the noise figure. Here, we use p_1 to denote -power and p_4 to denote -NF.

The performance data, shown in Fig. 9, are collected from the papers published at several major circuit conferences and journals, including JSSC, ISSCC, VLSI Symposium, RFIC Symposium, etc. These performance data cover a wide range of technology nodes, varying from 0.8 μm to 45 nm. In order to model the Pareto fronts of interest, we need to approximate three nonlinear functions: $p_2 = f_2(p_1)$, $p_3 = f_3(p_1, p_2)$, and $p_4 = f_4(p_1, p_2, p_3)$, as shown in (3).

Fig. 10 shows the modeling results for the 2-D Pareto front $p_2 = f_2(p_1)$ that captures the tradeoff between -power and gain. We set up the derivative constraints to enforce that the first-order derivative $\mathrm{d}p_2/\mathrm{d}p_1$ of gain with respect to -power is nonpositive. As a result, the Pareto front $p_2 = f_2(p_1)$ is monotonically decreasing, as shown in Fig. 10(a). The nonpositive derivative $\mathrm{d}p_2/\mathrm{d}p_1$ is further plotted in Fig. 10(b).

Fig. 11 shows the 3-D Pareto front $p_3 = f_3(p_1, p_2)$ that captures the tradeoff between -power, gain, and IIP3. The 4-D Pareto front $p_4 = f_4(p_1, p_2, p_3)$ for modeling the tradeoff between -power, gain, IIP3, and -NF is 3-D. We plot $f_4(\bullet)$ by using three 3-D plots as shown in Fig. 12. Due to the page limit, we do not plot the corresponding derivatives for $f_4(\bullet)$ here.

Fig. 13 shows the histograms of absolute modeling error for three nonlinear functions: $p_2 = f_2(p_1)$, $p_3 = f_3(p_1, p_2)$, and $p_4 = f_4(p_1, p_2, p_3)$. Table II summarizes the modeling statistics for these Pareto fronts. Note that the RMS error is less than 0.15 for all Pareto front models in this example. These results demonstrate the superior accuracy of our proposed algorithm for Pareto front modeling. The extracted Pareto front models can be used by analog designers for efficient system-level optimization.

VI. Conclusion

In this paper, we propose a novel technique to search the design knowledge from online resources, and then encode the knowledge as Pareto fronts. In order to accurately model high-dimensional Pareto fronts, an efficient sparse regression algorithm is adopted to automatically identify the important basis functions based on a limited set of Pareto optimal samples. In addition, a grid discretization method is used to solve the semi-infinite optimization problem in order to enforce a monotonic Pareto front within the performance space. Our numerical examples demonstrate that the proposed modeling technique can accurately capture 4-D Pareto fronts for large-scale, complex analog systems. The high-dimensional Pareto fronts derived from our proposed work can be further used to guide analog designers to make important decisions for system-level optimization.

Note that the computational cost of our proposed Pareto front modeling algorithm may quickly increase, as the dimensionality (i.e., the number of performance metrics) increases. In the case where the dimensionality is extremely high, we must choose a set of important basis functions from numerous (e.g., billions of) possible candidates and, hence, the numerical algorithm described in this paper may become computationally unaffordable. In our future research, we will further study efficient heuristics to address this dimensionality issue.

Furthermore, note that the performance data collected from the Internet could be noisy or even contain outliers. When these performance data are used for Pareto front modeling, it is crucial to automatically detect and remove the outliers so that they do not bias the Pareto front models. However, due to the inequality constraints (i.e., the boundary and derivative constraints) posed by our Pareto front modeling procedure, most conventional outlier detection algorithms (e.g., robust regression [42]) cannot be directly applied. Hence, developing an efficient and robust outliner detection/removal algorithm with consideration of inequality constraints is another important research problem that must be carefully addressed before the proposed Pareto front modeling algorithm is practically applicable. The aforementioned outlier detection and removal problem will be further explored in our future research.

Finally, in addition to building Pareto front models for analog integrated circuits, our proposed approach can be further applied to large-scale analog systems consisting of multiple chips and/or external discrete components (e.g., board-level voltage regulator implemented with discrete power electronics and on-chip microcontroller). Applying Pareto front models to board-level design optimization is another interesting research topic that we plan to pursue in the future.

References

[1] R. Phelps, M. Krasnicki, R. Rutenbar, R. Carley, and J. Hellums, "Anaconda: Simulation-based synthesis of analog circuits via stochastic patter search," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 19, no. 6, pp. 703–717, Jun. 2000.

[2] F. Schenkel *et al.*, "Mismatch analysis and direct yield optimization by spec-wise linearization and feasibility-guided search," in *Proc Design Autom. Conf.*, Las Vegas, NV, USA, 2001, pp. 858–863.

[3] M. Hershenson, S. Boyd, and T. Lee, "Optimal design of a CMOS op-amp via geometric programming," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 20, no. 1, pp. 1–21, Jan. 2001.

[4] T. McConaghy and G. Gielen, "Template-free symbolic performance modeling of analog circuits via canonical-form functions and genetic programming," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 28, no. 8, pp. 1162–1175, Aug. 2009.

[5] G. Stehr, H. Graeb, and K. Antreich, "Performance tradeoff analysis of analog circuits by normal-boundary intersection," in *Proc. Design Autom. Conf.*, Anaheim, CA, USA, Jun. 2003, pp. 958–963.

[6] B. Smedt and G. Gielen, "Watson: Design space boundary exploration and model generation for analog and RF IC design," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 22, no. 2, pp. 213–224, Feb. 2003.

[7] S. Tiwary, S. Velu, R. Rutenbar, and T. Mukherjee, "Pareto optimal modeling for efficient PLL optimization," *Nanotech*, vol. 2, pp. 195–198, May 2004.

[8] T. Eeckelaert, T. McConaghy, and G. Gielen,, "Efficient multiobjective synthesis of analog circuits using hierarchical Pareto-optimal performance hypersurfaces," in *Proc. Design Autom. Test Europe*, Munich, Germany, Mar. 2005, pp. 1070–1075.

[9] T. McConaghy, P. Palmers, G. Gielen, and M. Steyaert, "Simultaneous multi-topology multi-objective sizing across thousands of analog circuit topologies," in *Proc. Design Autom. Conf.*, San Diego, CA, USA, Jun. 2007, pp. 944–947.

[10] G. Yu and P. Li, "Yield-aware analog integrated circuit optimization using geostatistics motivated performance modeling," in *Proc. Int. Conf. Comput.-Aided Design*, San Jose, CA, USA, Nov. 2007, pp. 464–569.

[11] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," *IEEE Trans. Evol. Comput.*, vol. 3, no. 4, pp. 257–271, Nov. 1999.

[12] R. Rutenbar, G. Gielen, and J. R. Chowdhury, "Hierarchical modeling, optimization, and synthesis for system-level analog and RF design," *Proc. IEEE*, vol. 95, no. 3, pp. 640–669, Mar. 2007.

[13] R. Walden, "Analog-to-digital converter survey and analysis," *IEEE J. Sel. Areas Commun.*, vol. 17, no. 4, pp. 539–550, Apr. 1999.

[14] J. Zou, "Hierarchical optimization of large-scale analog mixed-signal circuits based on Pareto optimal fronts," Ph.D. dissertation, Inst. Electron. Design Autom., Tech. Univ. Munich, Munich, Germany, 2009.

[15] J. Tao *et al.*, "Toward efficient programming of reconfigurable radio frequency (RF) receivers," in *Proc. Asia South Pac. Design Autom. Conf.*, Singapore, Jan. 2014, pp. 256–261.

[16] A. Singhee, "PTrace: Derivative-free local tracing of bicriterial design tradeoffs," in *Proc. Int. Conf. Comput.-Aided Design*, San Jose, CA, USA, Nov. 2011, pp. 498–502.

[17] M. Velasco-Jimenez, R. Castro-Lopez, E. Roca, and F. Fernandez, "Implementation issues in the hierarchical composition of performance models of analog circuits," in *Proc. Design Autom. Test Europe Conf. Exhibit.*, Dresden, Germany, Mar. 2014, pp. 1–6.

[18] D. Luenberger, *Microeconomic Theory*. New York, NY, USA: McGraw-Hill, 1995.

[19] C. Manning, P. Raghavan, and H. Schutze, *An Introduction to Information Retrieval*. New York, NY, USA: Cambridge Univ. Press, 2008.

[20] X. Li, "Finding deterministic solution from underdetermined equation: Large-scale performance modeling of analog/RF circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 29, no. 11, pp. 1661–1668, Nov. 2010.

[21] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York, NY, USA: Springer, 2009.

[22] M. Tipping, "Sparse Bayesian learning and the relevance vector machine," *J. Mach. Learn. Res.*, vol. 1, pp. 211–244, Jun. 2001.

[23] R. Hettich and K. Kortanek, "Semi-infinite programming: Theory, methods, and applications," *SIAM Rev.*, vol. 35, no. 3, pp. 380–429, Sep. 1993.

[24] P. Malcovati *et al.*, "Behavioral modeling of switched-capacitor sigma–delta modulators," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 50, no. 3, pp. 352–364, Mar. 2003.
[25] C. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006.
[26] R. Feldman and J. Sanger, *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. New York, NY, USA: Cambridge Univ. Press, 2007.
[27] A. Li, *Handbook of SAS DATA Step Programming*. Boca Raton, FL, USA: CRC Press, 2013.
[28] G. Argyrous, *Statistics for Research: With A Guide to SPSS*. Thousand Oaks, CA, USA: Sage, 2011.
[29] M. Barbaro *et al.*, "Label-free, direct DNA detection by means of a standard CMOS electronic chip," *Sens. Actuat. B-Chem.*, vols. 171–172, pp. 148–154, Aug./Sep. 2012.
[30] X. Zou, X. Xu, L. Yao, and Y. Lian, "A 1-V 450-nW fully integrated programmable biomedical sensor interface chip," *IEEE J. Solid-State Circuits*, vol. 44, no. 4, pp. 1067–1077, Apr. 2009.
[31] V. Jain, S. Sundararaman, and P. Heydari, "A CMOS 22-29GHz receiver front-end for UWB automotive pulse-radars," in *Proc. IEEE Custom Integr. Circuits Conf.*, San Jose, CA, USA, 2007, pp. 757–760.
[32] J. De La Rosa *et al.*, "A CMOS 110-dB@ 40-kS/s programmable-gain chopper-stabilized third-order 2-1 cascade sigma-delta modulator for low-power high-linearity automotive sensor ASICs," *IEEE J. Solid-State Circuits*, vol. 40, no. 11, pp. 2246–2264, Nov. 2005.
[33] G. Qu and L. Yuan, "Design THINGS for the Internet of Things—An EDA perspective," in *Proc. IEEE Int. Conf. Comput.-Aided Design*, Piscataway, NJ, USA, 2014, pp. 411–416.
[34] D. Miorandi, S. Sicari, F. De Pellegrini, and I. Chlamtac, "Internet of things: Vision, applications and research challenges," *Ad Hoc Netw.*, vol. 10, no. 7, pp. 1497–1516, Sep. 2012.
[35] D. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.
[36] E. Candes, "Compressive sampling," in *Proc. Int. Congr. Math*, Madrid, Spain, Aug. 2006, pp. 1433–1452.
[37] D. Donoho and J. Tanner, "Precise undersampling theorems," *Proc. IEEE*, vol. 98, no. 6, pp. 913–924, Jun. 2010.
[38] H. Lin, P. Li, and C. Myers, "Verification of digitally-intensive analog circuits via kernel ridge regression and hybrid reachability analysis," in *Proc. IEEE Design Autom. Conf.*, Austin, TX, USA, May./Jun. 2013, pp. 1–6.
[39] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge Univ. Press, 2004.
[40] B. Razavi, *Design of Analog CMOS Integrated Circuits*. New York, NY, USA: McGraw-Hill, 2001.
[41] B. Murmann. (Mar. 2, 2014). *ADC Performance Survey 1997-2014.* [Online]. Available: http://web.stanford.edu/~murmann/adcsurvey.html
[42] P. Rousseeuw and A. Leroy, *Robust Regression and Outlier Detection*. New York, NY, USA: Wiley, 2003.

Jun Tao (M'07) received the B.S. and Ph.D. degrees in electrical engineering from Fudan University, Shanghai, China, in 2002 and 2007, respectively.

Since 2007, she has been an Assistant Professor with the Department of Microelectronics, Fudan University. She was a Visiting Scholar with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, USA, in 2012. Her current research interests include modeling, simulation, and design for analog and mixed-signal systems.

Changhai Liao received the B.S. degree from the School of Microelectronics and Solid-State Electronics, University of Electronic Science and Technology of China, Chengdu, China, in 2013. He is currently pursuing the Ph.D. degree in microelectronics and solid-state electronics with Fudan University, Shanghai, China.

His current research interests include statistical methods for computer-aided design under process variation and design automation for analog/mixed-signal integrated circuits.

Xuan Zeng (M'97) 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 Department of Microelectronics, Fudan University. She served as the Director of the State Key Laboratory of Application Specific Integrated Circuits and Systems, Fudan University 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 Department of Microelectronics, Technische Universiteit Delft, Delft, The Netherlands, in 2002 and 2003, respectively. Her current research interests include design for manufacturability, high-speed interconnect analysis and optimization, and circuit simulation.

Dr. Zeng was a recipient of the Chinese National Science Funds for Distinguished Young Scientists in 2011, the First-Class of Natural Science Prize of Shanghai in 2012, and the Changjiang Distinguished Professor from the Ministry of Education Department of China in 2014.

Xin Li (S'01–M'06–SM'10) received the B.S. and M.S. degrees in electronics engineering from Fudan University, Shanghai, China, in 1998 and 2001, respectively, and the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, USA, in 2005.

He is currently an Associate Professor with the Department of Electrical and Computer Engineering, Carnegie Mellon University. In 2005, he has co-founded Xigmix Inc., Pittsburgh, PA, USA, to commercialize his Ph.D. research, and served as the Chief Technical Officer until the company was acquired by Extreme DA, Santa Clara, CA, USA, in 2007. From 2009 to 2012, he was an Assistant Director with the FCRP Focus Research Center for Circuit and System Solutions (C2S2), a national consortium of 13 research universities (CMU, MIT, Stanford, Berkeley, UIUC, UMich, Columbia, UCLA, and among others) chartered by the U.S. Semiconductor Industry, and the U.S. Department of Defense to work on next-generation integrated circuit design challenges. From 2014 to 2015, he was appointed as an Assistant Director with the Center for Silicon System Implementation, Pittsburgh, PA, USA, a CMU Research Center. His current research interests include integrated circuit and signal processing.

Dr. Li was a recipient of the NSF Faculty Early Career Development Award in 2012, the IEEE Donald O. Pederson Best Paper Award in 2013, the Best Paper Award from Design Automation Conference (DAC), in 2010, two IEEE/ACM William J. McCalla International Conference on Computer-Aided Design (ICCAD) Best Paper Awards in 2004 and 2011, and the Best Paper Award from International Symposium on Integrated Circuits in 2014. He was an Associate Editor of *ACM Transaction on Design Automation of Electronic Systems*, the IEEE TRANSACTION ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, and the *Journal of Low Power Electronics*. He served on the ACM/SIGDA Outstanding Ph.D. Dissertation Award Selection Committee, the IEEE Outstanding Young Author Award Selection Committee, the Technical Program Committee of DAC, and the Technical Program Committee of ICCAD.