Finding Deterministic Solution from Underdetermined Equation: Large-Scale Performance Variability Modeling of Analog/RF Circuits

Xin Li, *Member, IEEE*

***Abstract*—The aggressive scaling of integrated circuit technology results in high-dimensional, strongly-nonlinear performance variability that cannot be efficiently captured by traditional modeling techniques. In this paper, we adapt a novel L_0-norm regularization method to address this modeling challenge. Our goal is to solve a large number of (e.g., 10^4–10^6) model coefficients from a small set of (e.g., 10^2–10^3) sampling points without overfitting. This is facilitated by exploiting the underlying sparsity of model coefficients. Namely, although numerous basis functions are needed to span the high-dimensional, strongly-nonlinear variation space, only a few of them play an important role for a given performance of interest. An efficient orthogonal matching pursuit (OMP) algorithm is applied to automatically select these important basis functions based on a limited number of simulation samples. Several circuit examples designed in a commercial 65 nm process demonstrate that OMP achieves up to 25× speedup compared to the traditional least-squares fitting method.**

***Index Terms*—Integrated circuit, performance modeling, process variation.**

I. Introduction

As INTEGRATED circuit (IC) technologies scale to 65 nm and beyond, process variation becomes increasingly critical and makes it continually more challenging to create a reliable, robust design with high yield [3]. For analog/mixed-signal circuits designed for sub-65 nm technology nodes, parametric yield loss due to manufacturing variation becomes a significant or even dominant portion of the total yield loss. Hence, process variation must be carefully considered within today's IC design flow.

Unlike most digital circuits that can be efficiently analyzed at gate level (e.g., by statistical timing analysis [4], [5]), analog/mixed-signal circuits must be modeled and simulated at transistor level. To estimate the performance variability of these circuits, response surface modeling (RSM) has been widely applied [6]–[13]. The objective of RSM is to approximate the circuit performance (e.g., delay, gain, and so on) as an analytical (either linear or nonlinear) function of device parameters (e.g., V_{TH}, T_{OX}, and so on). Once response surface models are created, they can be used for various purposes, e.g., efficiently predicting performance distributions [8].

While RSM was extensively studied in the past, the following two trends in advanced IC technologies suggest a need to revisit this area.

1) *Strong nonlinearity*: as process variation becomes relatively large, simple linear RSM is not sufficiently accurate [8]. Instead, nonlinear (e.g., quadratic) models are required to accurately predict performance variability.
2) *High dimensionality*: random device mismatch becomes increasingly important due to technology scaling [3]. To accurately model this effect, a large number of random variables must be utilized, rendering a high-dimensional variation space [9]–[13].

The combination of these two recent trends results in a large-scale RSM problem that is difficult to solve. For instance, as will be demonstrated in Section V, more than 10^4 independent random variables must be used to model the device-level variation of a simplified SRAM critical path designed in a commercial 65 nm CMOS process. To create a quadratic model for the critical path delay, we must determine a $10^4 \times 10^4$ quadratic coefficient matrix including 10^8 coefficients.

Most existing RSM techniques [9]–[13] rely on least-squares (LS) fitting. They solve model coefficients from an over-determined linear equation and, hence, the number of sampling points must be equal to or greater than the number of model coefficients. Since each sampling point is created by expensive transistor-level simulation, such high simulation cost prevents us from fitting high-dimensional, strongly-nonlinear models where a great number of sampling points are required. While the existing RSM techniques have been successfully applied to small-size or medium-size problems (e.g., 10–1000 model coefficients), they are ill-equipped to address the modeling needs of today's analog/mixed-signal system where 10^4–10^6 model coefficients must be solved. The

Manuscript received August 31, 2009; revised January 26, 2010 and May 13, 2010; accepted June 4, 2010. Date of current version October 20, 2010. This work was supported in part by the National Science Foundation, under Contract CCF-0811023, and by Mentor Graphics Corporation. This paper was presented in part at the IEEE/ACM Design Automation Conference in 2008 [1] and 2009 [2]. This paper was recommended by Associate Editor J. R. Phillips.

The author 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.2010.2061292

challenging issue is how to make RSM affordable for such a *large* problem size.

In this paper, we proposed a novel RSM technique that aims to solve a large number of (e.g., 10^4–10^6) model coefficients from a small set of (e.g., 10^2–10^3) sampling points without over-fitting. While numerous basis functions must be used to span the high-dimensional, strongly-nonlinear variation space, not all these functions play an important role for a given performance of interest. In other words, although there are a large number of unknown model coefficients, many of these coefficients are close to zero, rendering a unique *sparse* structure. Taking the 65 nm SRAM in Section V as an example, the delay variation of its critical path can be accurately approximated by around 40 basis functions, even though the SRAM circuit contains 21 310 independent random variables. However, we do not know the right basis functions in advance; these important basis functions must be automatically selected by a "smart" algorithm based on a limited number of simulation samples.

Our proposed RSM algorithm borrows the recent advance of statistics [14]–[19], [22], [27] to explore the underlying sparsity of model coefficients. It applies *L_0-norm regularization* [14] to find the unique sparse solution (i.e., the model coefficients) of an underdetermined equation. Importantly, several theoretical studies from the statistics community prove that with some general assumptions, the L_0-norm regularization approach guarantees to find all model coefficients with high accuracy [14]–[19].

An important contribution of this paper is to apply an efficient orthogonal matching pursuit (OMP) algorithm [19] to solve the L_0-norm regularization problem. For our RSM application, OMP empirically shows superior modeling accuracy over the statistical regression (STAR) algorithm proposed in [1] and the least angle regression (LAR) algorithm proposed in [2]. Compared to STAR and LAR, OMP reduces modeling error by 1.5–5× with negligible computational overhead, as will be demonstrated by the numerical examples in Section V.

The remainder of this paper is organized as follows. In Section II, we review the background on response surface modeling, and then describe the L_0-norm regularization scheme in Section III. The OMP algorithm is used to efficiently determine all model coefficients in Section IV. The efficacy of OMP is demonstrated by several numerical examples in Section V, followed by the conclusion in Section VI.

II. Background

Given N process parameters $X = [x_1\ x_2\ ...\ x_N]^{\mathrm{T}}$, the process variations $\Delta X = X - X_0$, where X_0 contains the mean values of X, are often modeled as the random variables that are jointly normal [6]–[13]. In such cases, principal component analysis (PCA) [20] can be applied to find a set of independent factors $\Delta Y = [\Delta y_1\ \Delta y_2\ ...\ \Delta y_N]^{\mathrm{T}}$ to represent the original correlated random variables. To analyze the variability of a circuit performance f, the following response surface model is used to approximate f as the linear combination of M basis functions [9]–[13], [21], [22], [27]:

$$f(\Delta Y) \approx \sum_{m=1}^{M} \alpha_m \cdot g_m(\Delta Y) \tag{1}$$

where $\{\alpha_m; m = 1, 2, ..., M\}$ are the model coefficients, and $\{g_m(\Delta Y); m = 1, 2, ..., M\}$ are the basis functions (e.g., linear and quadratic polynomials).

The performance function $f(\Delta Y)$ is a local perturbation of its nominal value. To approximate such a local variation effect, we apply polynomial basis functions in this paper, similar to other traditional techniques [9]–[12]. Without loss of generality, we further assume that the basis functions $\{g_m(\Delta Y); m = 1, 2, ..., M\}$ are normalized and orthogonal

$$\int_{-\infty}^{+\infty} g_i(\Delta Y) \cdot g_j(\Delta Y) \cdot pdf(\Delta Y) \cdot d(\Delta Y) = \begin{cases} 1 & (i = j) \\ 0 & (i \neq j) \end{cases} \tag{2}$$

where $pdf(\Delta Y)$ is the joint probability density function of ΔY. Since the random variables in ΔY are independent and standard normal after PCA, the basis functions $\{g_m(\Delta Y); m = 1, 2, ..., M\}$ can be found by the expansion of Hermite series [24] and, therefore, are referred to as Hermite polynomials. For example, if we consider the simple 1-D case, the Hermite polynomials can be expressed as [24]

$$g_1(\Delta y) = 1 \quad g_2(\Delta y) = \Delta y \quad g_3(\Delta y) = \frac{1}{\sqrt{2}} \cdot \left(\Delta y^2 - 1\right) \quad \cdots . \tag{3}$$

Extending (3) to the 2-D case yields [24]

$$\begin{array}{cc} g_1(\Delta y_1, \Delta y_2) = 1 & g_2(\Delta y_1, \Delta y_2) = \Delta y_1 \\ g_3(\Delta y_1, \Delta y_2) = \Delta y_2 & g_4(\Delta y_1, \Delta y_2) = \frac{1}{\sqrt{2}} \cdot \left(\Delta y_1^2 - 1\right) \\ g_5(\Delta y_1, \Delta y_2) = \Delta y_1 \Delta y_2 & \cdots \end{array} . \tag{4}$$

High-dimensional Hermite polynomials can also be constructed by using the formulas presented in [24]. The aforementioned representation of orthogonal polynomials facilitates us to develop an efficient numerical algorithm for response surface modeling, as will be discussed in detail in Section IV.

In general, the unknown model coefficients in (1) can be determined by solving the following linear equation at K sampling points:

$$\sum_{m=1}^{M} \alpha_m \cdot g_m\left(\Delta Y^{(k)}\right) = f^{(k)} \quad (k = 1, 2, \cdots, K) \tag{5}$$

where $\Delta Y^{(k)}$ and $f^{(k)}$ are the values of ΔY and $f(\Delta Y)$ at the kth sampling point, respectively. Equation (5) can be equivalently represented as the following matrix form:

$$\sum_{m=1}^{M} \alpha_m \cdot G_m = G \cdot \alpha = F \tag{6}$$

where

$$G_m = \left[\ g_m\left(\Delta Y^{(1)}\right) \quad g_m\left(\Delta Y^{(2)}\right) \quad \cdots \quad g_m\left(\Delta Y^{(K)}\right)\ \right]^T \tag{7}$$

$$G = \left[\ G_1 \quad G_2 \quad \cdots \quad G_M\ \right] \tag{8}$$

$$\alpha = \left[\ \alpha_1 \quad \alpha_2 \quad \cdots \quad \alpha_M\ \right]^T \tag{9}$$

$$F = \left[\ f^{(1)} \quad f^{(2)} \quad \cdots \quad f^{(K)}\ \right]^T . \tag{10}$$

In (6)–(10), the vector $G_m \in R^K$ contains the sampling points for the mth basis function $g_m(\Delta Y)$. It can be conceptually considered as the basis vector associated with $g_m(\Delta Y)$.

Most existing RSM techniques [9]–[13] attempt to solve the LS solution for (6). Hence, the number of samples (K) must be equal to or greater than the number of coefficients (M). It, in turn, becomes intractable, if M is large (e.g., 10^4–10^6). For this reason, the traditional RSM techniques are limited to small-size or medium-size problems (e.g., 10–1000 model coefficients). In this paper, we propose a novel RSM algorithm that aims to create high-dimensional, strongly-nonlinear response surface models (e.g., 10^4–10^6 model coefficients) from a small set of (e.g., 10^2–10^3) simulation samples without over-fitting.

III. L_0-Norm Regularization

Unlike the traditional RSM techniques that solve model coefficients from an over-determined equation, we focus on the non-trivial case where the number of samples (K) is less than the number of coefficients (M). Namely, there are fewer equations than unknowns, and the linear system in (6) is underdetermined. In this case, the solution α (i.e., the model coefficients) is not unique, unless additional constraints are added.

In this paper, we will explore the sparsity of α to uniquely determine its value. Our approach is motivated by the observation that while a large number of basis functions must be used to span the high-dimensional, strongly-nonlinear variation space, only a few of them are required to approximate a specific performance function. In other words, the vector α in (6) only contains a small number of non-zeros. However, we do not know the exact locations of these non-zeros. In what follows, we will utilize a novel L_0-norm regularization scheme [14]–[19], [22], [27] to find the non-zeros of α so that the solution of the underdetermined equation (6) can be uniquely solved.

To illustrate the idea of L_0-norm regularization, we formulate the following optimization to solve the sparse solution α for (6)

$$\begin{array}{ll} \underset{\alpha}{\text{minimize}} & \|G \cdot \alpha - F\|_2^2 \\ \text{subject to} & \|\alpha\|_0 \leq \lambda \end{array} \tag{11}$$

where $||\bullet||_2$ and $||\bullet||_0$ stand for the L_2-norm and L_0-norm of a vector, respectively. The L_0-norm $||\alpha||_0$ equals the number of non-zeros in the vector α. It measures the sparsity of α. Therefore, by directly constraining the L_0-norm, the optimization in (11) attempts to find a sparse solution α that minimizes the sum of squared residuals.

The parameter λ in (11) explores the tradeoff between the sparsity of the solution α and the minimal value of the cost function $||G \cdot \alpha - F||_2^2$. For instance, a large λ will result in a small cost function, but meanwhile it will increase the number of non-zeros in α. It is important to note that a small cost function does not necessarily mean a small modeling error. Even though the minimal cost function value can be reduced by increasing λ, such a strategy may result in over-fitting especially because (6) is underdetermined. In the extreme case, if λ is sufficiently large and the constraint in (11) is not active, we can always find a solution α to make the cost function exactly zero. However, such a solution is likely to be useless, since it over-fits the given sampling points. In practice, the optimal value of λ can be automatically determined by cross-validation, as will be discussed in detail in Section IV.

While the aforementioned L_0-norm regularization can effectively guarantee a sparse solution α, the optimization in (11) is non-deterministic polynomial-time (NP) hard [14]–[19], [22], [27] and, hence, is extremely difficult to solve. In what follows, we will describe an efficient heuristic algorithm to solve (11) using OMP [19].

IV. Orthogonal Matching Pursuit

Given the underdetermined linear equation (6), OMP [19] applies a heuristic algorithm to identify a small set of (say, λ) important basis functions and use them to approximate the performance function $f(\Delta Y)$. For other non-critical basis functions $g_m(\Delta Y)$'s, the corresponding coefficients α_m's are set to zero. If the number of selected basis functions (i.e., λ) is substantially less than the total number of basis functions (i.e., M), the resulting solution α is sparse. In this section, we describe the OMP algorithm at a level that is intuitive to the CAD community. More mathematical details of OMP can be found in [19].

A. Basis Function Selection

A critical component of the OMP algorithm is to identify a subset of important basis functions that significantly impact the performance function $f(\Delta Y)$. OMP uses the inner product between $f(\Delta Y)$ and $g_m(\Delta Y)$ to measure the importance of the basis function $g_m(\Delta Y)$

$$\langle f, g_m \rangle = \int_{-\infty}^{+\infty} f(\Delta Y) \cdot g_m(\Delta Y) \cdot pdf(\Delta Y) \cdot d(\Delta Y). \tag{12}$$

In other words, if a basis function $g_m(\Delta Y)$ is highly correlated with $f(\Delta Y)$, it has a strong impact on $f(\Delta Y)$.

Since the basis functions $\{g_m(\Delta Y);\ m = 1, 2, ..., M\}$ are normalized and orthogonal, the inner product defined in (12) is exactly equal to the model coefficient α_m

$$\begin{aligned} \langle f, g_m \rangle &= \int_{-\infty}^{+\infty} \left[\sum_{i=1}^{M} \alpha_i g_i(\Delta Y) \right] \cdot g_m(\Delta Y) \cdot pdf(\Delta Y) \cdot d(\Delta Y) \\ &= \sum_{i=1}^{M} \alpha_i \cdot \int_{-\infty}^{+\infty} g_i(\Delta Y) \cdot g_m(\Delta Y) \cdot pdf(\Delta Y) \cdot d(\Delta Y) \\ &= \alpha_m \end{aligned} \tag{13}$$

This is the reason why the inner product in (12) can be used as a good criterion to measure the importance of each basis function. Namely, if the inner product $\langle f, g_m \rangle$ (i.e., α_m) is far away from zero, the corresponding basis function $g_m(\Delta Y)$ should be selected to approximate the performance function $f(\Delta Y)$.

In practice, we do not know the analytical form of $f(\Delta Y)$ and, hence, the integration in (12) must be numerically computed from a set of sampling points. To this end, unlike the traditional response surface modeling techniques that generate sampling points by design of experiment [25], we randomly draw K sampling points $\{(\Delta Y^{(k)},\ f^{(k)});\ k = 1, 2, ..., K\}$ based on the probability density function $pdf(\Delta Y)$. The inner product $\langle f,\ g_m \rangle$ in (12) is *approximated* as [26]

$$\rho_m = \frac{1}{K} \cdot \sum_{k=1}^{K} f^{(k)} \cdot g_m\left(\Delta Y^{(k)}\right) = \frac{1}{K} \cdot G_m^T \cdot F \qquad (14)$$

where $G_m \in R^K$ and $F \in R^K$ are defined in (7) and (10), respectively.

According to (12) and (13), we notice that (14) is a statistic estimator of α_m, i.e., it gives an estimation of the unknown coefficient value α_m. Such an estimation, however, is not highly accurate, as the estimator ρ_m in (14) is calculated from random sampling data $\{(\Delta Y^{(k)},\ f^{(k)});\ k = 1, 2, ..., K\}$ that may contain large fluctuations [1], [26]. For this reason, the OMP algorithm does not simply use the estimator ρ_m in (14) to determine the value of the model coefficient α_m. Instead, the inner product estimated by (14) is only used to identify important basis functions and the model coefficients are consequently solved by least-squares fitting for these important basis functions. In addition, to further improve the accuracy of basis function selection, OMP applies an iterative algorithm to select a single most important basis function at each iteration step. This iterative algorithm will be discussed in detail in the next subsection.

B. Iterative Algorithm

Given the underdetermined linear equation (6), OMP iteratively selects the important basis functions based on the criterion shown in Section IV-A. It calculates the inner product values $\{\rho_m;\ m = 1, 2, ..., M\}$ as defined in (14), and then find the basis vector G_{s1} [or equivalently, the basis function $g_{s1}(\Delta Y)$] that is most correlated with F, i.e., $|\rho_{s1}|$ takes the largest value. Note that only a single basis vector is selected at this moment. Once G_{s1} is identified, OMP approximates F in the direction of G_{s1}

$$F \approx \alpha_{s1} \cdot G_{s1} \qquad (15)$$

where the coefficient α_{s1} is determined by solving the following least-squares fitting problem:

$$\underset{\alpha_{s1}}{\text{minimize}} \quad \|\alpha_{s1} \cdot G_{s1} - F\|_2^2 \ . \qquad (16)$$

Next, OMP removes the component $\alpha_{s1} \times G_{s1}$ from F and calculates the residual

$$Res = F - \alpha_{s1} \cdot G_{s1}. \qquad (17)$$

Based on (17), OMP further identifies the next important basis vector G_{s2} by calculating the inner product values between the residual Res and all basis vectors $\{G_m; m = 1, 2, ..., M\}$

$$\xi_m = \frac{1}{K} \cdot G_m^T \cdot Res. \qquad (18)$$

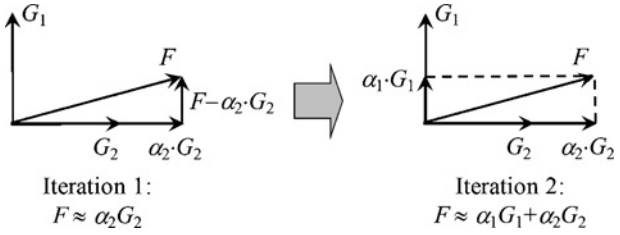

Fig. 1. OMP calculates the model coefficients α_1 and α_2 for a 2-D example: $F = \alpha_1 G_1 + \alpha_2 G_2$.

Once G_{s2} is known, OMP approximates F in the directions of both G_{s1} and G_{s2}

$$F \approx \alpha_{s1} \cdot G_{s1} + \alpha_{s2} \cdot G_{s2}. \qquad (19)$$

In (19), the coefficients α_{s1} and α_{s2} are found by solving the following optimization problem:

$$\underset{\alpha_{s1},\alpha_{s2}}{\text{minimize}} \quad \|\alpha_{s1} \cdot G_{s1} + \alpha_{s2} \cdot G_{s2} - F\|_2^2 \ . \qquad (20)$$

It is important to note that α_{s1} calculated by (16) may be different from that calculated by (20). In other words, every time when a new basis function is selected, OMP re-calculates all model coefficients to minimize the sum of squared residuals. This re-calculation step is required, because even though the basis functions $\{g_m(\Delta Y;\ m = 1, 2, ..., M\}$ are orthogonal as defined in (2), the basis vectors $\{G_m; m = 1, 2, ..., M\}$ are not necessarily orthogonal, i.e., $G_i^T \cdot G_j \neq 0\,(i \neq j)$, due to random sampling. Hence, the new basis function selected at the current iteration step may change the model coefficient values calculated at previous iteration steps.

The aforementioned iteration for basis function selection and least-squares fitting continues until a sufficient number of (i.e., λ) important basis vectors are identified. Algorithm 1 summarizes the major iteration steps of OMP.

To intuitively understand the OMP algorithm, we consider the 2-D example shown in Fig. 1. In this example, there are two basis vectors G_1 and G_2. The vector G_2 has a stronger correlation with F than the vector G_1. Hence, G_2 is first selected to approximate F, i.e., $F \approx \alpha_2 \cdot G_2$, where α_2 is determined by least-squares fitting. The residual of the approximation is $Res = F - \alpha_2 \cdot G_2$, which is orthogonal to the basis vector G_2, i.e., $G_2^T \cdot Res = 0$.

Next, the inner product values between the residual Res and both basis vectors G_1 and G_2 are calculated. Since $G_2^T \cdot Res$ is equal to zero, the vector G_1 is now selected and F is approximated as $F \approx \alpha_1 \cdot G_1 + \alpha_2 \cdot G_2$, where α_1 and α_2 are calculated to minimize the sum of squared residuals: $||\alpha_1 \cdot G_1 + \alpha_2 \cdot G_2 - F||_2^2$. In this example, since only two basis vectors G_1 and G_2 are used to span the 2-D space, OMP stops at the second iteration step. If more than two basis vectors are involved, OMP will continue to add extra basis vectors to the "most important" set until the termination criterion is satisfied.

It is important to note that even though OMP is a heuristic algorithm to solve the L_0-norm regularization problem in (11), the quality of its solution is guaranteed according to several theoretical studies from the statistics community [19]. Roughly speaking, if the M-dimensional vector α contains P non-zeros

Algorithm 1: Orthogonal Matching Pursuit (OMP)

1. Start from the linear equation $G \cdot \alpha = F$ in (6) and a given integer number λ representing the total number of basis vectors that should be selected.
2. Set the residual $Res = F$, the basis vector set $\Omega = \{\}$, and the iteration index $p = 1$.
3. Calculate the inner product values $\{\xi_m; m = 1, 2, ..., M\}$ between Res and all basis vectors $\{G_m; m = 1, 2, ..., M\}$ using (18).
4. Select the basis vector G_s that has the largest $|\xi_s|$.
5. Update Ω by $\Omega = \Omega \cup \{s\}$.
6. Approximate F by the linear combination of $\{G_i; i \in \Omega\}$, i.e., the important basis vectors that are already selected:
$$F \approx \sum_{i \in \Omega} \alpha_i \cdot G_i \tag{21}$$
where the model coefficients are determined by least-squares fitting
$$\underset{\alpha_i, i \in \Omega}{\text{minimize}} \quad \left\| \sum_{i \in \Omega} \alpha_i \cdot G_i - F \right\|_2^2 . \tag{22}$$
7. Calculate the residual:
$$Res = F - \sum_{i \in \Omega} \alpha_i \cdot G_i. \tag{23}$$
8. If $p < \lambda$, $p = p + 1$ and go to Step 3. Otherwise, go to Step 9.
9. For any G_i that is not selected (i.e., $i \notin \Omega$), the corresponding coefficient α_i is set to 0.

($P << M$) and the linear equation $G \cdot \alpha = F$ in (6) is well-conditioned, the actual solution α can be *almost* uniquely determined (with a probability nearly equal to one) from K sampling points, where K is in the order of $O(P \cdot \log M)$ [19]. While this theoretical result does not precisely give the number of required sampling points, it presents an important scaling trend. Namely, K (the number of sampling points) is a logarithmic function of M (the number of unknown coefficients). It, in turn, provides the theoretical foundation that by solving the sparse solution of an underdetermined equation, a large number of model coefficients can be uniquely determined from a small number of sampling points.

C. Cross-Validation

The OMP algorithm (i.e., Algorithm 1) relies on a user-defined λ, i.e., the total number of basis functions that should be selected. In practice, λ is not known in advance. The appropriate value of λ must be determined by considering the following two important issues. First, if λ is too small, OMP will not select a sufficient number of basis functions to approximate the performance function $f(\Delta Y)$, thereby leading to large modeling error. On the other hand, if λ is too large and OMP uses too many basis functions to approximate $f(\Delta Y)$, it will result in over-fitting which again prevents us from extracting an accurate performance model. Hence, in order to achieve the best modeling accuracy, we must accurately estimate the modeling error for different λ values and then find the optimal λ to minimize modeling error.

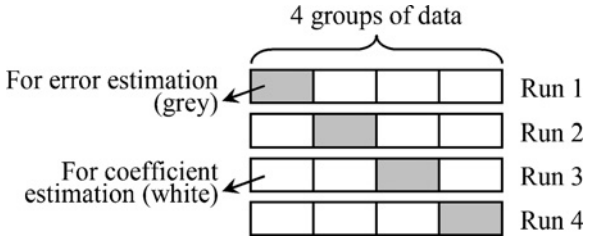

Fig. 2. 4-Fold cross-validation partitions the data set into four groups and modeling error is estimated from four independent runs.

However, given a limited number of sampling points, accurately estimating modeling error is not a trivial task. To avoid over-fitting, we cannot simply measure the modeling error from the same sampling data that are used to calculate the model coefficients. Instead, modeling error must be measured from an independent data set. Cross-validation is an efficient method for model validation that has been widely used in the statistics community [22], [27]. A Q-fold cross-validation partitions the entire data set into Q groups, as shown by the example in Fig. 2. Modeling error is estimated from Q independent runs. In each run, one of the Q groups is used to estimate the modeling error and all other groups are used to calculate the model coefficients. Different groups should be selected for error estimation in different runs. As such, each run results in an error value ε_q ($q = 1, 2, ..., Q$) that is measured from a unique group of sampling points. In addition, when a model is trained and tested in each run, non-overlapped data sets are used so that over-fitting can be easily detected. The final modeling error is computed as the average of $\{\varepsilon_q; q = 1, 2, ..., Q\}$, i.e., $\varepsilon = (\varepsilon_1 + \varepsilon_2 + ... + \varepsilon_Q)/Q$.

In our case, the OMP algorithm is used to iteratively select important basis functions and calculate model coefficients for different λ values during each cross-validation run. Next, the modeling error associated with each run is estimated, resulting in $\{\varepsilon_q(\lambda);\ q = 1, 2, ..., Q\}$. Note that ε_q is not simply a value, but a 1-D function of λ. Once all cross-validation runs are complete, the final modeling error is calculated as $\varepsilon(\lambda) = [\varepsilon_1(\lambda) + \varepsilon_2(\lambda) + ... + \varepsilon_Q(\lambda)]/Q$, which is again a 1-D function of λ. The optimal λ is then determined by finding the minimal value of ε (λ).

The major drawback of cross-validation is the need to repeatedly extract the model coefficients for Q times. However, for our circuit modeling application, the overall computational cost is dominated by the transistor-level simulation that is required to generate sampling data. Hence, the computational overhead by cross-validation is almost negligible, as will be demonstrated by our numerical examples in Section V.

V. Numerical Examples

In this section, we demonstrate the efficacy of OMP using several circuit examples designed in a commercial 65 nm process. For each example, two independent random sampling sets, called training set and testing set respectively, are generated using Cadence Spectre. The training set is used for coefficient fitting (including cross-validation), while the testing set is used for model validation. All numerical experiments are performed on a 2.8 GHz Linux server.

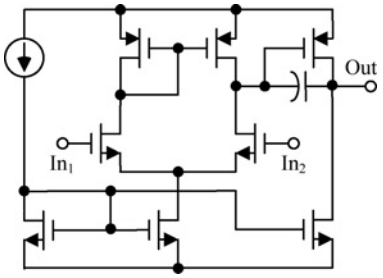

Fig. 3. Simplified circuit schematic of a two-stage operational amplifier.

For testing and comparison, four different performance modeling techniques are implemented: LS fitting [21], STAR [1], LAR [2], and the proposed OMP. LS aims to solve an over-determined linear equation in (6) by minimizing the sum of squared residuals. STAR is similar to OMP. It applies the same inner product criterion to iteratively select the important basis functions. However, unlike Step 6 of Algorithm 1, STAR directly uses the inner product in (18) to determine the model coefficient of the selected basis function at each iteration step. Finally, LAR relaxes the L_0-norm $\|\alpha\|_0$ in (11) by L_1-norm $\|\alpha\|_1$, i.e., the summation of the absolute values of all elements in α. After $\|\alpha\|_0$ is replaced by $\|\alpha\|_1$, (11) can be re-formulated as a convex optimization problem and efficiently solved by an iterative algorithm that is referred to as least angle regression in [16].

A. Two-Stage Operational Amplifier

Fig. 3 shows the simplified circuit schematic of a two-stage operational amplifier (OpAmp) that contains an on-chip current source for biasing. In this example, we aim to model four performance metrics: "gain," "bandwidth," "power," and "offset," considering both inter-die and intra-die variations of MOS transistors and layout parasitics. After PCA based on foundry data, 630 independent random variables are extracted to model these variations.

1) *Linear Performance Modeling:* Fig. 4 shows the error for four different modeling techniques: LS fitting [21], STAR [1], LAR [2], and the proposed OMP. To achieve the same accuracy, STAR, LAR, and OMP require much less training samples than LS, because they solve the unknown model coefficients from an underdetermined equation by exploiting the underlying sparsity of model coefficients. In this example, such a sparse structure exists, since the variability of each circuit-level performance metric is dominated by a few device-level variation sources. For instance, the offset of the OpAmp is mainly determined by the device mismatches of the input differential pair in Fig. 3.

Studying Fig. 4, we would notice that STAR, LAR, and OMP yield different modeling accuracy, given the same number of training samples. Even though all these three modeling techniques build sparse performance models, they rely on different algorithms to select the important basis functions and/or determine the model coefficients, as mentioned at the beginning of this section. In this example, OMP offers better accuracy (up to 1.5–5× error reduction) than STAR, as shown in Fig. 4. Remember that once an important basis function is

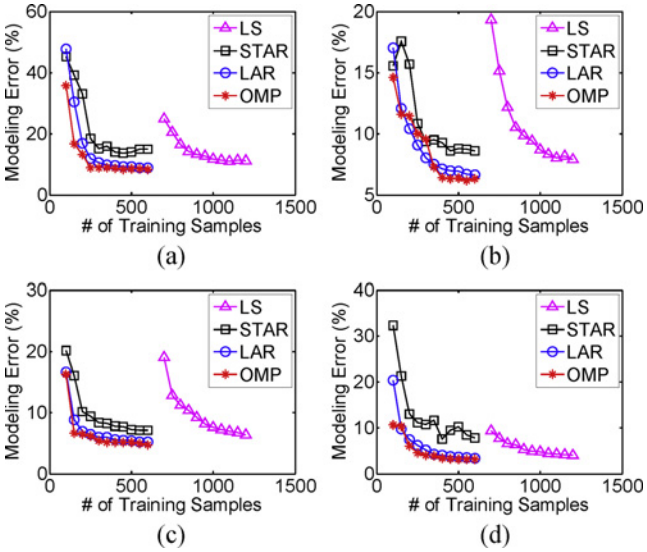

Fig. 4. Linear modeling error decreases, as the number of training samples increases. (a) Gain. (b) Bandwidth. (c) Power. (d) Offset.

TABLE I
LINEAR PERFORMANCE MODELING COST FOR OPERATIONAL AMPLIFIER

	LS [21]	STAR [1]	LAR [2]	OMP
# of training samples	1200	600	600	600
Simulation cost (s)	16 140	8070	8070	8070
Fitting cost (s)	2.6	1.2	44.2	26.4
Total cost (s)	16 142	8071	8114	8096

selected by STAR, it simply uses the inner product in (18) to determine the corresponding model coefficient. On the other hand, OMP optimally solves the unknown model coefficients of all selected basis functions by least-squares fitting, as shown in Step 6 of Algorithm 1. For this reason, even though both STAR and OMP apply the same inner product criterion for basis function selection, OMP results in superior modeling accuracy over STAR.

Compared to LAR, OMP also shows slightly improved modeling accuracy in most cases. However, there are a few examples where LAR outperforms OMP, as shown in Fig. 4(b). In general, LAR and OMP apply different heuristics to solve the L_0-norm regularization problem in (11). To the best of our knowledge, there is no theoretical evidence to prove that one method is always better than the other.

Table I summarizes the modeling cost for all four modeling techniques: LS, STAR, LAR, and OMP. The overall computational cost for performance modeling consists of two portions: 1) simulation cost (i.e., the cost of running a transistor-level simulator to generate all sampling points in the training set), and 2) fitting cost (i.e., the cost of solving all model coefficients from the sampling points). For our circuit modeling application, the computational cost is dominated by transistor-level simulation. Therefore, even though OMP takes more time to fit all model coefficients than LS, it still achieves 2× runtime speedup over LS in this example.

2) *Quadratic Performance Modeling:* To further improve modeling accuracy, we select 200 most important device-level

TABLE II
QUADRATIC PERFORMANCE MODELING ERROR FOR OPERATIONAL AMPLIFIER

	LS [21]	STAR [1]	LAR [2]	OMP
Gain	4.21%	8.03%	5.77%	4.39%
Bandwidth	3.84%	5.36%	4.11%	2.94%
Power	1.52%	4.37%	1.69%	1.17%
Offset	3.69%	9.15%	2.94%	1.88%

TABLE III
QUADRATIC PERFORMANCE MODELING COST FOR OPERATIONAL AMPLIFIER

	LS [21]	STAR [1]	LAR [2]	OMP
# of training samples	25 000	1000	1000	1000
Simulation cost (s)	336 250	13 450	13 450	13 450
Fitting cost (s)	51 562	92	1449	1174
Total cost (s)	387 812	13 542	14 899	14 624

process parameters based on the magnitude of the linear model coefficients. Next, we create quadratic performance models using these critical process parameters. In this example, the 200-dimensional quadratic model contains 20 301 unknown coefficients. Tables II and III show the accuracy and cost for the aforementioned quadratic modeling, respectively. As shown in Table II, OMP reduces the modeling error by 1.5–3×, compared to STAR and LAR. In addition, compared to LS, OMP reduces the computational time from 4 days to 4 h (24× speedup) while achieving similar accuracy, as shown in Table III.

In this example, even though there are 20 301 basis functions in total, OMP automatically selects less than 100 important basis functions for all performance functions (in particular, 88 basis functions for "gain," 95 basis functions for "bandwidth," 96 basis functions for "power," and 98 basis functions for "offset"). It, in turn, implies that the quadratic performance modeling problem studied in this example is profoundly sparse. Such a sparse structure is the necessary condition to make OMP feasible and efficient in this example.

B. SRAM Read Path

Shown in Fig. 5 is the simplified circuit schematic of an SRAM read path that contains cell array, replica path for self-timing, and sense amplifier. In this example, both inter-die and intra-die variations are considered. After PCA based on foundry data, 21 310 independent random variables are extracted to model these variations.

We aim to model the read delay from the word line (WL) to the sense amplifier output (Out). Since the read delay is primarily determined by the transistors and interconnects on the read path, we expect to observe a sparse structure for the delay model in this example. Namely, a large number of model coefficients will be close to zero, if the corresponding basis functions are associated with the local device mismatches outside the read path. Hence, the SRAM circuit in Fig. 5 offers a good example for us to test the efficacy of the proposed performance modeling technique.

For testing and comparison, Table IV shows the linear performance modeling error and cost for four different techniques:

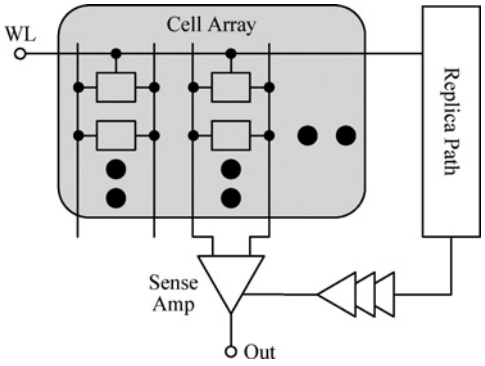

Fig. 5. Simplified circuit schematic of an SRAM read path.

TABLE IV
LINEAR PERFORMANCE MODELING ERROR AND COST FOR SRAM READ PATH

	LS [21]	STAR [1]	LAR [2]	OMP
Modeling error	9.78%	6.34%	4.94%	4.09%
# of training samples	25 000	1000	1000	1000
Simulation cost (s)	728 250	29 130	29 130	29 130
Fitting cost (s)	13856.1	26.5	338.3	169.7
Total cost (s)	742 106	29 156	29 468	29 300

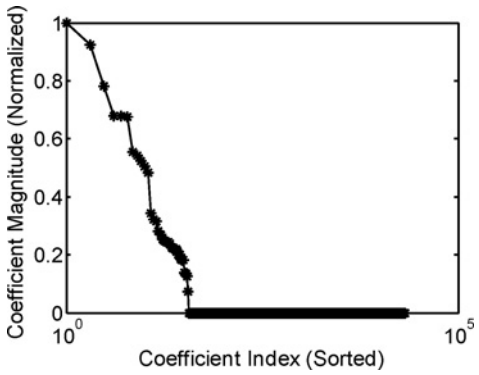

Fig. 6. Large number of model coefficients of SRAM read delay are close to zero (estimated by OMP).

LS fitting, STAR, LAR, and the proposed OMP. As shown in Table IV, OMP is most accurate among these four methods. In addition, compared to LS, OMP reduces the computational time from 8.5 days to 8.2 h (25× speedup).

Fig. 6 further shows the magnitude of the linear model coefficients estimated by OMP. Studying Fig. 6, we would notice that even though there are 21 311 basis functions in total, only 36 basis functions are selected and their corresponding model coefficients are non-zero. These 36 basis functions are automatically identified by OMP to accurately capture the delay variation. This sparse structure is the essential necessary condition that makes the proposed OMP technique applicable to this example.

VI. CONCLUSION

In this paper, we proposed a novel L_0-norm regularization scheme to efficiently create high-dimensional linear and

nonlinear performance models for nanoscale circuits. The proposed method was facilitated by exploiting the unique sparse structure of model coefficients. An efficient OMP algorithm was used to solve the proposed L_0-norm regularization problem. Several numerical examples demonstrated that, compared to least-squares fitting, OMP achieves up to 25× runtime speedup without surrendering any accuracy. Furthermore, compared to the STAR algorithm proposed in [1] and the LAR algorithm proposed in [2], OMP reduces modeling error by 1.5–5× with negligible computational overhead for our tested examples. However, we should point out that LAR and OMP apply different heuristics to solve the L_0-norm regularization problem in (11). To the best of our knowledge, there is no theoretical evidence to prove that one method is always better than the other.

REFERENCES

[1] X. Li and H. Liu, "Statistical regression for efficient high-dimensional modeling of analog and mixed-signal performance variations," in *Proc. Des. Autom. Conf.*, 2008, pp. 38–43.

[2] X. Li, "Finding deterministic solution from underdetermined equation: Large-scale performance modeling by least angle regression," in *Proc. Des. Autom. Conf.*, 2009, pp. 364–369.

[3] Semiconductor Industry Associate. (2007). *International Technology Roadmap for Semiconductors* [Online]. Available: www.itrs.net/Links/2007ITRS/Home2007.htm

[4] H. Chang and S. Sapatnekar, "Statistical timing analysis under spatial correlations," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 24, no. 9, pp. 1467–1482, Sep. 2005.

[5] C. Visweswariah, K. Ravindran, K. Kalafala, S. Walker, S. Narayan, D. Beece, J. Piaget, N. Venkateswaran, and J. Hemmett, "First-order incremental block-based statistical timing analysis," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 25, no. 10, pp. 2170–2180, Oct. 2006.

[6] A. Dharchoudhury and S. Kang, "Worse-case analysis and optimization of VLSI circuit performance," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 14, no. 4, pp. 481–492, Apr. 1995.

[7] F. Schenkel, M. Pronath, S. Zizala, R. Schwencker, H. Graeb, and K. Antreich, "Mismatch analysis and direct yield optimization by spec-wise linearization and feasibility-guided search," in *Proc. Des. Autom. Conf.*, 2001, pp. 858–863.

[8] X. Li, J. Le, P. Gopalakrishnan, and L. Pileggi, "Asymptotic probability extraction for non-normal performance distributions," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 26, no. 1, pp. 16–37, Jan. 2007.

[9] X. Li, J. Le, L. Pileggi, and A. Strojwas, "Projection-based performance modeling for inter/intra-die variations," in *Proc. Int. Conf. Comput.-Aided Design*, 2005, pp. 721–727.

[10] Z. Feng and P. Li, "Performance-oriented statistical parameter reduction of parameterized systems via reduced rank regression," in *Proc. Int. Conf. Comput.-Aided Design*, 2006, pp. 868–875.

[11] A. Singhee and R. Rutenbar, "Beyond low-order statistical response surfaces: Latent variable regression for efficient, highly nonlinear fitting," in *Proc. Design Autom. Conf.*, 2007, pp. 256–261.

[12] A. Mitev, M. Marefat, D. Ma, and J. Wang, "Principle Hessian direction based parameter reduction for interconnect networks with process variation," in *Proc. Int. Conf. Comput.-Aided Design*, 2007, pp. 632–637.

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

[14] R. Tibshirani, "Regression shrinkage and selection via the Lasso," *J. Royal Statist. Soc.*, vol. 58, no. 1, pp. 267–288, 1996.

[15] S. Chen, D. Donoho, and M. Saunders, "Atomic decomposition by basis pursuit," *SIAM Rev.*, vol. 43, no. 1, pp. 129–159, 2001.

[16] B. Efron, T. Hastie, and I. Johnstone, "Least angle regression," *Ann. Statist.*, vol. 32, no. 2, pp. 407–499, 2004.

[17] E. Candes, "Compressive sampling," in *Proc. Int. Congr. Math.*, 2006, pp. 1433–1452.

[18] D. Donoho, "Compressed sensing," *IEEE Trans. Inform. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.

[19] J. Tropp and A. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. Information Theory*, vol. 53, no. 12, pp. 4655–4666, Dec. 2007.

[20] G. Seber, *Multivariate Observations* (Wiley Series). New York: Wiley, 1984.

[21] R. Myers and D. Montgomery, *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*. New York: Wiley-Interscience, 2002.

[22] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. Berlin, Germany: Springer, 2003.

[23] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, MA: Cambridge University Press, 2004.

[24] G. Sansone, *Orthogonal Functions*. New York: Dover, 2004.

[25] D. Montgomery, *Design and Analysis of Experiments*. New York: Wiley, 2005.

[26] C. Robert and G. Casella, *Monte Carlo Statistical Methods*. Berlin, Germany: Springer, 2005.

[27] C. Bishop, *Pattern Recognition and Machine Learning*. Berlin, Germany: Springer, 2006.

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

He is currently an Assistant Professor with the Department of Electrical and Computer Engineering, CMU. In 2005, he co-founded Xigmix, Inc., Pittsburgh, to commercialize his Ph.D. research, and served as the Chief Technical Officer until the company was acquired in 2007. Since 2009, he has been the Assistant Director for the FCRP Focus Research Center for Circuit and System Solutions, a national consortium of 13 research universities (CMU, Massachusetts Institute of Technology, Cambridge, Stanford, Berkeley, University of Illinois at Urbana-Champaign, Urbana, University of Michigan, Ann Arbor, Columbia, University of California, Los Angeles, 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. His current research interests include computer-aided design and neural signal processing.

Dr. Li served on the Technical Program Committee of the International Conference on Computer-Aided Design from 2008 to 2010, the Technical Program Committee of the International Workshop on Timing Issues in the Specification and Synthesis of Digital Systems in 2010, the Technical Program Committee of the International Conference on VLSI Design in 2009, the Technical Program Committee of the International Conference on Image Theory and Applications in 2009, and the IEEE Outstanding Young Author Award Selection Committee in 2006. He received the IEEE/ACM William J. McCalla ICCAD Best Paper Award in 2004, the Best Paper Nomination from the Design Automatic Conference in 2006 and 2010, and the Best Session Award from the Semiconductor Research Corporation Student Symposium in 2006. He received the Inventor Recognition Award from the Focus Center Research Program in 2006, 2007, and 2009.