Finding Deterministic Solution from Underdetermined Equation: Large-Scale Performance Modeling by Least Angle Regression

Xin Li
ECE Department, Carnegie Mellon University
5000 Forbs Avenue, Pittsburgh, PA 15213
xinli@ece.cmu.edu

ABSTRACT

The aggressive scaling of IC 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_1-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 over-fitting. 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 algorithm of least angle regression (LAR) is applied to automatically select these important basis functions based on a limited number of simulation samples. Several circuit examples designed in a commercial 65nm process demonstrate that LAR achieves up to 25× speedup compared with the traditional least-squares fitting.

Categories and Subject Descriptors

B.7.2 [**Integrated Circuits**]: Design Aids – Verification

General Terms

Algorithms

Keywords

Process Variation, Response Surface Modeling

1. INTRODUCTION

As IC technologies scale to 65nm and beyond, process variation becomes increasingly critical and makes it continually more challenging to create a reliable, robust design with high yield [1]. For analog/mixed-signal circuits designed for sub-65nm 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), 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 [2]-[6]. The objective of RSM is to approximate the circuit performance (e.g., delay, gain, etc.) as an analytical (either linear or nonlinear) function of device parameters (e.g., V_{TH}, T_{OX}, etc.). Once response surface models are created, they can be used for various purposes, e.g., efficiently predicting performance distributions [7].

DAC'09, July 26-31, 2009, San Francisco, California, USA

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

- **Strong nonlinearity**: As process variation becomes relatively large, simple linear RSM is not sufficiently accurate [7]. Instead, nonlinear (e.g., quadratic) models are required to accurately predict performance variability.
- **High dimensionality**: Random device mismatch becomes increasingly important due to technology scaling [1]. To accurately model this effect, a large number of random variables must be utilized, rendering a high-dimensional variation space [6].

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 5, 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 65nm 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 [2]-[5] rely on least-squares (LS) fitting. They solve model coefficients from an over-determined 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, 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 challenging issue is how to make RSM affordable for such a *large* problem size.

In this paper, we propose 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 65nm SRAM in Section 5 as an example, the delay variation of its critical path can be accurately approximated by around 50 basis functions, even though the SRAM circuit contains 21310 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 [8] to explore the underlying sparsity of model

coefficients. It applies *L_1-norm regularization* [8] to find the unique sparse solution (i.e., the model coefficients) of an underdetermined equation. Importantly, the proposed L_1-norm regularization is formulated as a convex optimization problem and, therefore, can be solved both robustly and efficiently.

An important contribution of this paper is to apply an efficient algorithm of *Least Angle Regression* (LAR [8]) to solve the L_1-norm regularization problem. For our RSM application, LAR is substantially more efficient than the well-known interior-point method [13] that was developed for general-purpose convex optimization. In addition, LAR results in more accurate response surface models than the statistical regression (STAR) algorithm proposed in [6], which can be proven by both theoretical analyses [10] and numerical experiments. Compared with STAR, LAR reduces modeling error by 1.5~3× with negligible computational overhead, as will be demonstrated by the numerical examples in Section 5.

The remainder of this paper is organized as follows. In Section 2, we review the background on principal component analysis and response surface modeling, and propose our L_1-norm regularization in Section 3. The LAR algorithm is used to efficiently solve all model coefficients in Section 4. The efficacy of LAR is demonstrated by several numerical examples in Section 5, followed by the conclusions in Section 6.

2. BACKGROUND

2.1 Principal Component Analysis

Given N process parameters $X = [x_1\ x_2\ ...\ x_N]^T$, the process variation $\Delta X = X - X_0$, where X_0 denotes the mean value of X, is often modeled by multiple zero-mean, correlated Normal distributions [2]-[7]. Principal component analysis (PCA) [11] is a statistical method that finds a set of independent factors to represent the correlated Normal distributions. Assume that the correlation of ΔX is represented by a symmetric, positive semi-definite covariance matrix R. PCA decomposes R as [11]:

$$R = U \cdot \Sigma \cdot U^T \tag{1}$$

where $\Sigma = diag(\lambda_1, \lambda_2, ..., \lambda_N)$ contains the eigenvalues of R, and $U = [U_1\ U_2\ ...\ U_N]$ contains the corresponding eigenvectors that are orthonormal, i.e., $U^T U = I$. (I is the identity matrix.) PCA defines a set of new random variables $\Delta Y = [\Delta y_1\ \Delta y_2\ ...\ \Delta y_N]^T$:

$$\Delta Y = \Sigma^{-0.5} \cdot U^T \cdot \Delta X\ . \tag{2}$$

The new random variables in ΔY are called the principal components. It is easy to verify that all principal components in ΔY are independent and standard Normal (i.e., zero mean and unit variance). More details on PCA can be found in [11].

2.2 Response Surface Modeling

Given a circuit design, the circuit performance f (e.g., delay, gain, etc.) is a function of the process variation ΔY defined in (2). RSM approximates the performance function $f(\Delta Y)$ as the linear combination of M basis functions [2]-[6]:

$$f(\Delta Y) \approx \sum_{i=1}^{M} \alpha_i \cdot g_i(\Delta Y) \tag{3}$$

where $\{\alpha_i;\ i = 1,2,...,M\}$ are the model coefficients, and $\{g_i(\Delta Y);\ i = 1,2,...,M\}$ are the basis functions (e.g., linear, quadratic, etc.). The unknown model coefficients in (3) can be determined by solving the following linear equation at K sampling points:

$$G \cdot \alpha = F \tag{4}$$

where

$$G = \begin{bmatrix} g_1(\Delta Y^{(1)}) & g_2(\Delta Y^{(1)}) & \cdots & g_M(\Delta Y^{(1)}) \\ g_1(\Delta Y^{(2)}) & g_2(\Delta Y^{(2)}) & \cdots & g_M(\Delta Y^{(2)}) \\ \vdots & \vdots & \vdots & \vdots \\ g_1(\Delta Y^{(K)}) & g_2(\Delta Y^{(K)}) & & g_M(\Delta Y^{(K)}) \end{bmatrix} \tag{5}$$

$$\alpha = [\alpha_1 \quad \alpha_2 \quad \cdots \quad \alpha_M]^T \tag{6}$$

$$F = [f^{(1)} \quad f^{(2)} \quad \cdots \quad f^{(K)}]^T . \tag{7}$$

In (5)-(7), $\Delta Y^{(k)}$ and $f^{(k)}$ are the values of ΔY and $f(\Delta Y)$ at the k-th sampling point respectively. Without loss of generality, we assume that all basis functions are normalized:

$$G_i^T G_i = 1 \quad (i = 1,2,\cdots,M) \tag{8}$$

where

$$G_i = [g_i(\Delta Y^{(1)}) \quad g_i(\Delta Y^{(2)}) \quad \cdots \quad g_i(\Delta Y^{(K)})]^T . \tag{9}$$

This assumption simplifies the notation of our discussion in the following sections.

Most existing RSM techniques [2]-[5] attempt to solve the least-squares (LS) solution for (4). 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.

3. L_1-NORM REGULARIZATION

Our proposed RSM technique utilizes a novel L_1-norm regularization scheme that is derived from advanced statistics theories [8]. In this section, we describe its mathematical formulation and highlight the novelties.

3.1 Mathematical Formulation

Unlike the traditional RSM techniques that solve model coefficients from an over-determined equation, we focus on the nontrivial 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 (4) 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, 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. We will propose a novel L_1-norm regularization scheme to find the non-zeros so that the solution α of the underdetermined equation (4) can be uniquely solved.

To derive the proposed L_1-norm regularization, we first show the idea of L_0-norm regularization. To this end, we formulate the following optimization to solve the sparse solution α for (4):

$$\begin{aligned} \underset{\alpha}{\text{minimize}} \quad & \|G \cdot \alpha - F\|_2^2 \\ \text{subject to} \quad & \|\alpha\|_0 \le \lambda \end{aligned} \tag{10}$$

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 (10) attempts to find a sparse solution α that minimizes the least-squares error.

The parameter λ in (10) 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 Eq. (4) is underdetermined. In the extreme case, if λ is sufficiently large and the constraint in (10) 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 4.

While the L_0-norm regularization can effectively guarantee a sparse solution α, the optimization in (10) is NP hard [8] and, hence, is extremely difficult to solve. A more efficient technique to find sparse solution is based on L_1-norm regularization – a relaxed version of L_0-norm:

$$\begin{aligned} \underset{\alpha}{\text{minimize}} \quad & \left\|G\cdot\alpha-F\right\|_2^2 \\ \text{subject to} \quad & \left\|\alpha\right\|_1 \le \lambda \end{aligned} \tag{11}$$

where $\|\alpha\|_1$ denotes the L_1-norm of the vector α, i.e., the summation of the absolute values of all elements in α:

$$\left\|\alpha\right\|_1 = \left|\alpha_1\right| + \left|\alpha_2\right| + \cdots + \left|\alpha_M\right|. \tag{12}$$

The L_1-norm regularization in (11) can be re-formulated as a convex optimization problem. Introduce a set of slack variables $\{\beta_i;\ i = 0,1,...,M\}$ and re-write (11) into the following equivalent form [13]:

$$\begin{aligned} \underset{\alpha,\beta}{\text{minimize}} \quad & \left\|G\cdot\alpha-F\right\|_2^2 \\ \text{subject to} \quad & \beta_1+\beta_2+\cdots+\beta_M \le \lambda \\ & -\beta_i \le \alpha_i \le \beta_i \qquad (i=1,2,\cdots,M) \end{aligned} \tag{13}$$

In (13), the cost function is quadratic and positive semi-definite. Hence, it is convex. All constraints are linear and, therefore, the resulting constraint set is a convex polytope. For these reasons, the L_1-norm regularization in (13) is a convex optimization problem and it can be solved by various efficient and robust algorithms, e.g., the interior-point method [13].

The aforementioned L_1-norm regularization is much more computationally efficient than the L_0-norm regularization that is NP hard. This is the major motivation to replace L_0-norm by L_1-norm. In the next sub-section, we will use a two-dimensional example to intuitively explain why solving the L_1-norm regularization in (11) yields a sparse solution α.

3.2 Geometrical Explanation

To understand the connection between L_1-norm regularization and sparse solution, we consider the two-dimensional example (i.e., $\alpha = [\alpha_1\ \alpha_2]^T$) in Fig. 1. Since the cost function $\|G\cdot\alpha-F\|_2^2$ is quadratic and positive semi-definite, its contour lines can be represented by multiple ellipsoids. On the other hand, the constraint $\|\alpha\|_1 \le \lambda$ corresponds to a number of rotated squares, associated with different values of λ. For example, two of such squares are shown in Fig. 1, where $\lambda_1 \le \lambda_2$.

Studying Fig. 1, we would notice that if λ is large (e.g., $\lambda = \lambda_2$), both α_1 and α_2 are not zero. However, as λ decreases (e.g., $\lambda = \lambda_1$), the contour of $\|G\cdot\alpha-F\|_2^2$ eventually intersects the polytope $\|\alpha\|_1 \le \lambda$ at one of its vertex. It, in turn, implies that one of the coefficients (i.e., α_1 in this case) becomes exactly zero. From this point of view, by decreasing λ of the L_1-norm regularization in (11), we can pose a strong constraint for sparsity and force a sparse solution. This intuitively explains why L_1-norm regularization guarantees sparsity, as is the case for L_0-norm regularization.

In addition, various theoretical studies from the statistics community demonstrate that under some general assumptions, both L_1-norm regularization and L_0-norm regularization result in the same solution [9]. Roughly speaking, if the M-dimensional vector α contains L non-zeros and the linear equation $G\cdot\alpha = F$ is well-conditioned, the solution α can be uniquely determined by L_1-norm regularization from K sampling points, where K is in the order of $O(L\cdot\log M)$ [9]. Note that K (the number of sampling points) is a logarithm 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.

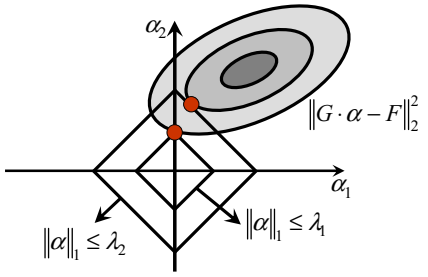


Fig. 1. The proposed L_1-norm regularization $\|\alpha\|_1 \le \lambda$ results in a sparse solution (i.e., $\alpha_1 = 0$) if λ is sufficiently small (i.e., $\lambda = \lambda_1$).

4. LEAST ANGLE REGRESSION

While Eq. (11) gives the mathematical formulation of L_1-norm regularization, a number of implementation issues must be carefully considered to make it of practical utility. Most importantly, an efficient algorithm is required to automatically determine the optimal value of λ. Towards this goal, a two-step approach can be used: (a) solve the optimization in (11) for a set of different λ's, and (b) select the optimal λ by cross-validation. In this section, we will introduce an efficient algorithm of least angle regression (LAR [8]) to accomplish these two steps with minimal computational cost.

4.1 Piece-wise Linear Solution Trajectory

To solve the optimization in (11) for different λ's, one straightforward approach is to repeatedly apply the interior-point method [13] to solve the convex programming problem in (13). This approach, however, is computationally expensive, as we must run a convex solver for many times in order to visit a sufficient number of possible values of λ. Instead of applying the interior-point method, we propose to first explore the unique property of the L_1-norm regularization in (11) and minimize the number of the possible λ's that we must visit.

As discussed in Section 3.2, the sparsity of the solution α depends on the value of λ. In the extreme case, if λ is zero, all coefficients in α are equal to zero. As λ gradually increases, more and more coefficients in α become non-zero. In fact, it can be proven that the solution α of (11) is a piece-wise linear function of

λ [8]. To intuitively illustrate this concept, we consider the following simple example:

$$f(\Delta Y) = -0.43 \cdot \Delta y_1 - 1.66 \cdot \Delta y_2 + 0.12 \cdot \Delta y_3 + 0.28 \cdot \Delta y_4 - 1.14 \cdot \Delta y_5 \quad (14)$$

We collected 50 random sampling points for this function and solved the L_1-norm regularization in (11) to calculate the values of α associated with different λ's. Fig. 2 shows the solution trajectory $\alpha(\lambda)$, i.e., α as a function of λ, which is piece-wise linear. The details of the mathematical proof for this piece-wise linear property can be found in [8].

The aforementioned piece-wise linear property allows us to find the entire solution trajectory $\alpha(\lambda)$ with low computational cost. We do not have to repeatedly solve the L_1-norm regularization at many different λ's. Instead, we only need to estimate the local linear function in each interval $[\lambda_i, \lambda_{i+1}]$. Next, we will show an iterative algorithm to efficiently find the solution trajectory $\alpha(\lambda)$.

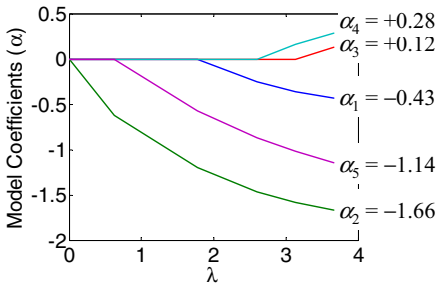


Fig. 2. The solution trajectory $\alpha(\lambda)$ of the L_1-norm regularization in (11) is a piece-wise linear function of λ.

4.2 Iterative Algorithm

We start from the extreme case where λ is zero for the L_1-norm regularization in (11). In this case, the solution of (11) is trivial: $\alpha = 0$. Our focus of this sub-section is to present an efficient algorithm of least angle regression (LAR) to calculate the solution trajectory $\alpha(\lambda)$, as λ increases from zero. To this end, we re-write the linear equation $G \cdot \alpha = F$ in (4)-(9) as:

$$F = \alpha_1 G_1 + \alpha_2 G_2 + \cdots + \alpha_M G_M . \quad (15)$$

Eq. (15) represents the vector F (i.e., the performance values) as the linear combination of the vectors $\{G_i;\ i = 1,2,...,M\}$ (i.e., the basis function values). Each G_i corresponds to a basis function $g_i(\Delta Y)$. As λ increases from zero, LAR [8] first calculates the correlation between F and every G_i:

$$r_i = \left| G_i^T F \right| \quad (i = 1,2,\cdots,M) \quad (16)$$

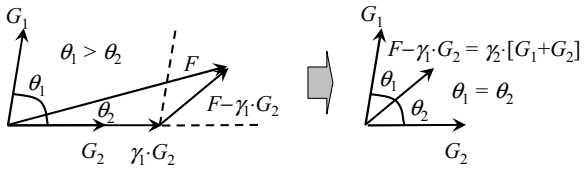
where G_i is a unit-length vector (i.e., $G_i^T G_i = 1$) as shown in (8). Next, LAR finds the vector G_{s1} that is most correlated with F, i.e., r_{s1} takes the largest value. Once G_{s1} is identified, LAR approximates F in the direction of G_{s1}:

$$F \approx \gamma_1 G_{s1} . \quad (17)$$

At this first iteration step, since we only use the basis function $g_{s1}(\Delta Y)$ to approximate the performance function $f(\Delta Y)$, the coefficients for all other basis functions (i.e., $\{\alpha_i;\ i \neq s1\}$) are zero. The residual of the approximation is:

$$Res = F - \gamma_1 G_{s1} . \quad (18)$$

To intuitively understand the LAR algorithm, we consider the two-dimensional example shown in Fig. 3. In this example, the vector G_2 has a higher correlation with F than the vector G_1. Hence, G_2 is selected to approximate F, i.e., $F \approx \gamma_1 G_2$. From the geometrical point of view, finding the largest correlation is equivalent to finding the least angle between the vectors $\{G_i;\ i = 1,2,...,M\}$ and the performance F. Therefore, the aforementioned algorithm is referred to as least angle regression in [8].



Iteration 1: $F \approx \alpha_2 G_2$ where $\alpha_2 = \gamma_1$

Iteration 2: $F \approx \alpha_1 G_1 + \alpha_2 G_2$ where $\alpha_1 = \gamma_2$ & $\alpha_2 = \gamma_1 + \gamma_2$

Fig. 3. LAR calculates the solution trajectory $\alpha(\lambda)$ of a two-dimensional example $F = \alpha_1 G_1 + \alpha_2 G_2$.

As $|\gamma_1|$ increases, the correlation between the vector G_{s1} and the residual $Res = F - \gamma_1 G_{s1}$ decreases. LAR uses an efficient algorithm to compute the maximal value of $|\gamma_1|$ at which the correlation between G_{s1} and $F - \gamma_1 G_{s1}$ is no longer dominant. In other words, there is another vector G_{s2} that has the same correlation with the residual:

$$\left| G_{s1}^T \cdot (F - \gamma_1 G_{s1}) \right| = \left| G_{s2}^T \cdot (F - \gamma_1 G_{s1}) \right| . \quad (19)$$

At this point, instead of continuing along G_{s1}, LAR proceeds in a direction equiangular between G_{s1} and G_{s2}. Namely, it approximates F by the linear combination of G_{s1} and G_{s2}:

$$F \approx \gamma_1 G_{s1} + \gamma_2 \cdot (G_{s1} + G_{s2}) \quad (20)$$

where the coefficient γ_1 is fixed at this second iteration step.

Taking Fig. 3 as an example, the residual $F - \gamma_1 G_2$ is approximated by $\gamma_2 \cdot (G_1 + G_2)$. If $|\gamma_2|$ is sufficiently large, F is exactly equal to $F = \gamma_2 G_1 + (\gamma_1 + \gamma_2) \cdot G_2$. In this example, because only two basis functions $g_1(\Delta Y)$ and $g_2(\Delta Y)$ are used, LAR stops at the second iteration step. If more than two basis functions are involved, LAR will keep increasing $|\gamma_2|$ until a third vector G_{s3} earns its way into the "most correlated" set, and so on. Algorithm 1 summarizes the major iteration steps of LAR.

Algorithm 1: Least Angle Regression (LAR)

1. Start from the vector F defined in (7) and the normalized vectors $\{G_i;\ i = 1,2,...,M\}$ defined in (8)-(9).
2. Apply (16) to calculate the correlation $\{r_i;\ i = 1,2,...,M\}$.
3. Select the vector G_s that has the largest correlation r_s.
4. Let the set $Q = \{G_s\}$ and the iteration index $p = 1$.
5. Approximate F by:
$$F \approx \gamma_p \cdot \sum_{G_i \in Q} G_i . \quad (21)$$
6. Calculate the residual:
$$Res = F - \gamma_p \cdot \sum_{G_i \in Q} G_i . \quad (22)$$
7. Use the algorithm in [8] to determine the maximal $|\gamma_p|$ such that either the residual in (22) equals 0 or another vector G_{new} ($G_{new} \notin Q$) has as much correlation with the residual:
$$\left| G_{new}^T \cdot Res \right| = \left| G_i^T \cdot Res \right| \quad (\forall G_i \in Q) . \quad (23)$$
8. If $Res = 0$, stop. Otherwise, $Q = Q \cup \{G_{new}\}$, $F = Res$, $p = p+1$, and go to Step 5.

It can be proven that with several small modifications, LAR will generate the entire piece-wise linear solution trajectory $\alpha(\lambda)$ for the L_1-norm regularization in (11) [8]. The computational cost of LAR is similar to that of applying the interior-point method to solve a *single* convex optimization in (11) with a fixed λ value.

Therefore, compared to the simple approach that repeatedly solves (11) for multiple λ's, LAR typically achieves orders of magnitude more efficiency, as is demonstrated in [8].

4.3 Cross-Validation

Once the solution trajectory $\alpha(\lambda)$ is extracted, we need to further find the optimal λ that minimizes the modeling error. 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 [12]. An S-fold cross-validation partitions the entire data set into S groups, as shown by the example in Fig. 4. Modeling error is estimated from S independent runs. In each run, one of the S 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_i (i = 1,2,...,S) 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 $\{\varepsilon_i;\ i = 1,2,...,S\}$, i.e., $\varepsilon = (\varepsilon_1+\varepsilon_2+...+\varepsilon_S)/S$.

For our application, LAR is used to calculate the solution trajectory during each cross-validation run. Next, the modeling error associated with each run is estimated, resulting in $\{\varepsilon_i(\lambda);\ i = 1,2,...,S\}$. Note that ε_i is not simply a value, but a one-dimensional function of λ. Once all cross-validation runs are complete, the final modeling error is calculated as $\varepsilon(\lambda) = (\varepsilon_1(\lambda)+\varepsilon_2(\lambda)+...+\varepsilon_S(\lambda))/S$, which is again a one-dimensional function of λ. The optimal λ is then determined by finding the minimal value of $\varepsilon(\lambda)$.

The major drawback of cross-validation is the need to repeatedly extract the model coefficients for S 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 5.

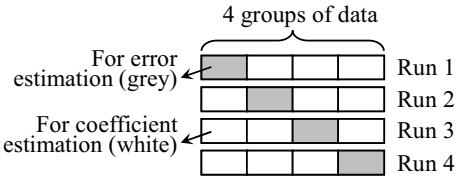


Fig. 4. A 4-fold cross-validation partitions the data set into 4 groups and modeling error is estimated from 4 independent runs.

5. NUMERICAL EXAMPLES

In this section we demonstrate the efficacy of LAR using several circuit examples. 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 2.8GHz Linux server.

5.1 Two-Stage Operational Amplifier

Fig. 5 shows the simplified circuit schematic of a two-stage operational amplifier (OpAmp) designed in a commercial 65nm process. In this example, we aim to model four performance metrics: gain, bandwidth, offset and power. The inter-die/intra-die variations of both MOS transistors and layout parasitics are considered. After PCA based on foundry data, 630 independent random variables are extracted to model these variations.

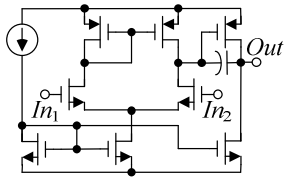


Fig. 5. Simplified circuit schematic of an operational amplifier.

A. Linear Performance Modeling

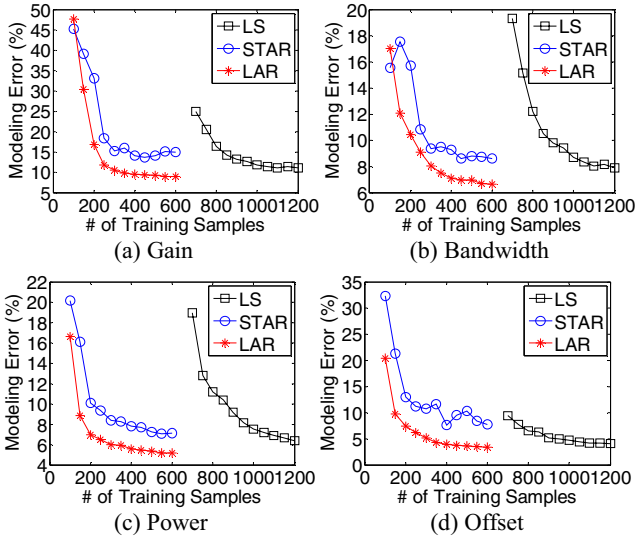


Fig. 6. Modeling error decreases as the number of training samples increases.

Table 1. Linear performance modeling cost for OpAmp

	LS	STAR	LAR
# of Samples	1200	600	600
Spectre (Sec.)	16140	8070	8070
Fitting (Sec.)	2.6	1.2	44.2
Total (Sec.)	16142	8071	8114

Fig. 6 shows the modeling error for three different techniques: least-squares fitting (LS), STAR [6] and LAR. To achieve the same accuracy, both STAR and LAR require much less training samples than LS, because they do not solve the unknown model coefficients from an over-determined equation. On the other hand, given the same number of training samples, LAR yields better accuracy (up to 2~3× error reduction) than STAR. STAR is similar to the orthogonal matching pursuit algorithm developed for signal processing [10]. It has been theoretically proven that the L_1-norm regularization used by LAR is more accurate, but also more expensive, than the orthogonal matching pursuit used by STAR [10]. However, for our circuit modeling application, the overall modeling cost is dominated by the Spectre simulation time that is required to generate sampling points. Therefore, the computational overhead of LAR is negligible, as shown in Table 1. LAR achieves 2× speedup compared with LS in this example.

B. Quadratic Performance Modeling

To further improve accuracy, we select 200 most important process parameters based on the magnitude of the linear model coefficients. Next, we create quadratic performance models for these critical process parameters. In this example, the 200-dimensional quadratic model contains 20301 unknown coefficients. Compared with STAR, LAR reduces the modeling error by 1.5~3×, as shown in Table 2. In addition, compared with LS, LAR reduces the modeling time from 4 days to 4 hours (24× speedup) while achieving similar accuracy, as shown in Table 3.

Table 2. Quadratic performance modeling error for OpAmp

	LS	STAR	LAR
Gain	4.21%	8.03%	5.77%
Bandwidth	3.84%	5.36%	4.11%
Power	1.52%	4.37%	1.69%
Offset	3.69%	9.15%	2.94%

Table 3. Quadratic performance modeling cost for OpAmp

	LS	STAR	LAR
# of Samples	25000	1000	1000
Spectre (Sec.)	336250	13450	13450
Fitting (Sec.)	51562	92	1449
Total (Sec.)	387812	13542	14899

5.2 Simplified SRAM Read Path

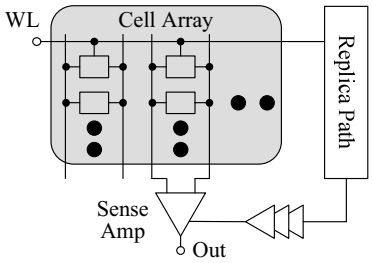


Fig. 7. Simplified circuit schematic of an SRAM read path.

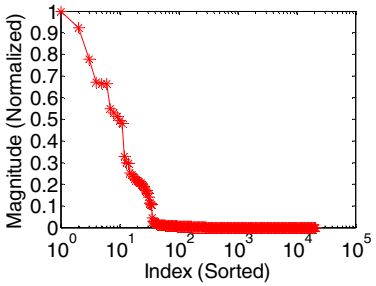


Fig. 8. Magnitude of the model coefficients estimated by LAR.

Table 4. Linear performance modeling error and cost for SRAM

	LS	STAR	LAR
Modeling Error	9.78%	6.34%	4.94%
# of Samples	25000	1000	1000
Spectre (Sec.)	728250	29130	29130
Fitting (Sec.)	13856.1	26.5	338.3
Total (Sec.)	742106	29156	29468

Shown in Fig. 7 is the simplified circuit schematic of an SRAM read path designed in a commercial 65nm process. The read path contains cell array, replica path for self-timing and sense amplifier. In this example, the performance of interest is the delay from the word line (WL) to the sense amplifier output (Out). Both inter-die and intra-die variations are considered. After PCA based on foundry data, 21310 independent random variables are extracted to model these variations.

Three different techniques are implemented for linear performance modeling: least-squares fitting (LS), STAR [6] and LAR. As shown in Table 4, LAR is most accurate among these three methods. Compared with LS, LAR reduces the modeling time from 8.6 days to 8.2 hours (25× speedup). Fig. 8 shows the magnitude of the linear model coefficients estimated by LAR. Even though there are 21311 basis functions in total, only 50 model coefficients are not close to zero. These 50 basis functions are automatically selected by LAR to accurately approximate the performance of interest in this example.

6. CONCLUSIONS

In this paper, we propose a novel L_1-norm regularization to efficiently create high-dimensional linear/nonlinear performance models for nanoscale circuits. The proposed method is facilitated by exploring the unique sparse structure of model coefficients. An efficient algorithm of least angle regression (LAR) is used to solve the proposed L_1-norm regularization problem. Our numerical examples demonstrate that compared with least-square fitting, LAR achieves up to 25× runtime speedup without surrendering any accuracy. LAR can be incorporated into a robust circuit design flow for efficient yield prediction and optimization.

7. ACKNOWLEDGEMENTS

This work has been supported by the National Science Foundation under contract CCF–0811023.

8. REFERENCES

[1] Semiconductor Industry Associate, *International Technology Roadmap for Semiconductors*, 2007.

[2] X. Li, J. Le, L. Pileggi and A. Strojwas, "Projection-based performance modeling for inter/intra-die variations," *IEEE ICCAD*, pp. 721-727, 2005.

[3] Z. Feng and P. Li, "Performance-oriented statistical parameter reduction of parameterized systems via reduced rank regression," *IEEE ICCAD*, pp. 868-875, 2006.

[4] A. Singhee and R. Rutenbar, "Beyond low-order statistical response surfaces: latent variable regression for efficient, highly nonlinear fitting," *IEEE DAC*, pp. 256-261, 2007.

[5] A. Mitev, M. Marefat, D. Ma and J. Wang, "Principle Hessian direction based parameter reduction for interconnect networks with process variation," *IEEE ICCAD*, pp. 632-637, 2007.

[6] X. Li and H. Liu, "Statistical regression for efficient high-dimensional modeling of analog and mixed-signal performance variations," *IEEE DAC*, pp. 38-43, 2008.

[7] X. Li, J. Le, P. Gopalakrishnan and L. Pileggi, "Asymptotic probability extraction for non-normal distributions of circuit performance," *IEEE ICCAD*, pp. 2-9, 2004.

[8] B. Efron, T. Hastie and I. Johnstone, "Least angle regression," *The Annals of Statistics*, vol. 32, no. 2, pp. 407-499, 2004.

[9] E. Candes, "Compressive sampling," *International Congress of Mathematicians*, 2006.

[10] 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, 2007.

[11] G. Seber, *Multivariate Observations*, Wiley Series, 1984.

[12] T. Hastie, R. Tibshirani and J. Friedman, *The Elements of Statistical Learning*, Springer, 2003.

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