Large-Scale Statistical Performance Modeling of Analog and Mixed-Signal Circuits

Xin Li, Wangyang Zhang and Fa Wang

Electrical & Computer Engineering Department, Carnegie Mellon University, Pittsburgh, PA 15213

Abstract—The aggressive scaling of IC technology results in large-scale performance variations that cannot be efficiently captured by traditional modeling techniques. This paper presents the recent development of statistical performance modeling and its important applications. In particular, we focus on two core techniques, sparse regression (SR) and Bayesian model fusion (BMF), that facilitate large-scale performance modeling with low computational cost. The basic ideas of SR and BMF are first explained and then their efficacy is compared to other traditional modeling approaches by using several analog and mixed-signal circuit examples.

I. Introduction

The aggressive technology scaling results in large-scale process variations that have been widely observed for today's integrated circuits (ICs) [1]-[3]. Analog and mixed-signal circuits are often sensitive to process variations, and parametric yield loss due to large-scale process variations has become a significant or even dominant portion of total yield loss for nanoscale analog and mixed-signal designs. Hence, appropriately modeling, analyzing and optimizing circuit variability is a critical task to ensure high product yield and, hence, low manufacturing cost today.

Towards this goal, most IC foundries have successfully developed statistical device models. In these models, the device-level parameters (e.g., V_{TH}, T_{OX}, etc.) are modeled as a set of random variables to account for process variations. To further study the performance variations at circuit level, statistical performance modeling has been extensively applied [4]-[20]. The objective of circuit-level performance modeling is to approximate the circuit performance (e.g., delay, gain, etc.) as an analytical (i.e., either linear or nonlinear) function of device-level parameters. Once the performance models are available, they can be used for various purposes such as critical variation source identification, parametric yield estimation and optimization [12]-[18], application-specific worst-case corner extraction [19]-[20], etc.

While statistical performance modeling has been extensively studied during the past several decades, the following two emerging trends posed by advanced IC technologies suggest an immediate need to revisit this area.

- **High dimensionality**: Random device mismatches have become increasingly important due to technology scaling [3]. To accurately model device mismatches, a large number of random variables must be utilized, rendering a high-dimensional variation space.
- **Strong nonlinearity**: As process variations become relatively large, simple linear performance models are not sufficiently accurate [2]. Instead, nonlinear (e.g., quadratic) models are required to accurately capture circuit-level performance variations.

The combination of these two recent trends leads to a large-scale performance modeling problem that is difficult to solve. For instance, as will be demonstrated in this paper, more than 10^4 independent random variables must be used to model the device-level variations of a simplified SRAM read path. To create a quadratic model for the read path delay, we must determine a $10^4 \times 10^4$ quadratic coefficient matrix including 10^8 coefficients!

This paper presents the recent development of statistical performance modeling and its important applications. In particular, we focus on two core techniques, *sparse regression* (SR) and *Bayesian model fusion* (BMF), that facilitate large-scale performance modeling with low computational cost. SR explores the fact that even though numerous device-level random variables must be used to capture the high-dimensional variation space, not all these random variables play an important role for a given circuit-level performance of interest. In other words, while there are a large number of unknown model coefficients associated with the device-level random variables, many of these model coefficients are close to zero, thereby rendering a unique sparse pattern.

On the other hand, BMF attempts to further reduce the modeling cost through a different avenue. It uses the simulation data generated at an early design stage to learn a model template that can be used to build the performance model at a late stage. For instance, we can extract a model template from schematic-level simulation data and then calibrate the model coefficients of the template with very few post-layout simulation samples. By "fusing" the schematic-level performance model with the post-layout performance model through the common template, the computational cost for post-layout performance modeling can be substantially reduced.

The remainder of this paper is organized as follows. In Section II, we first review the background on statistical performance modeling, and then describe the two core modeling techniques, SR and BMF, in Section III-IV. Several important applications for analog/mixed-signal performance modeling are further discussed in Section V. Finally, we conclude in Section VI.

II. Statistical Performance Modeling

A. Device-level Variation Modeling

According to their geometrical scales, device-level process variations can be classified into four different levels: lot-to-lot, wafer-to-wafer (within-lot), die-to-die (within-wafer) and

978-1-4673-1556-2/12/$31.00@2012 IEEE

within-die [21]. The first three variations are also called *inter-die variations*. They model the common variations across the die. The within-die variations are also called *intra-die variations* or *on-chip variations*. They model the individual local variations within the same die. Intra-die variations further consist of two different components: spatially-correlated variations and independent mismatches. The spatial correlation of intra-die variations is distance-dependent. A simple distance-based correlation model was developed by Pelgrom to capture the statistics of intra-die variations in [22]. Fig. 1 summarizes the aforementioned hierarchical structure of device-level process variations.

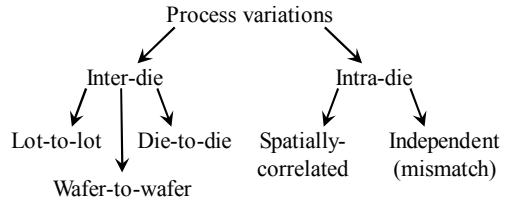

Fig. 1. A hierarchical structure is shown for device-level process variations according to their geometrical scales.

Without loss of generality, we consider M devices (e.g., transistors) in a circuit and N device-level parameters (e.g., V_{TH}) associated with each device. In this case, there are $M \cdot N$ device-level parameters in total and we represent them as an MN-dimensional vector

$$\mathbf{x} = [x_1 \quad x_2 \quad \cdots \quad x_{MN}]^T . \tag{1}$$

The process variations

$$\Delta\mathbf{x} = \mathbf{x} - \mathbf{x}_0 , \tag{2}$$

where $\mathbf{x}_0$ denotes the mean value of $\mathbf{x}$, are often modeled as a number of random variables that are zero mean and jointly Normal [2].

The random variables in $\mathbf{x}$ are typically correlated due to the following two reasons. First, the same variation source may affect multiple device-level parameters and, hence, make them correlated. For instance, the variation of V_{TH} is partially caused by the variation of T_{OX}. For this reason, the two random variables ΔV_{TH} and ΔT_{OX} are partially correlated. Second, due to the spatial correlation of process variations, the device-level parameters associated with different devices can be correlated. This is another important reason why correlated device-level random variables are observed.

Given a set of correlated random variables defined in (2), principal component analysis (PCA) [23] can be applied to find a set of independent random variables

$$\Delta\mathbf{y} = \mathbf{P} \cdot \Delta\mathbf{x} , \tag{3}$$

where

$$\Delta\mathbf{y} = [\Delta y_1 \quad \Delta y_2 \quad \cdots \quad \Delta y_{MN}]^T , \tag{4}$$

to represent the original correlated random variables. In (3), the linear transformation matrix $\mathbf{P}$ is determined such that the random variables in $\Delta\mathbf{y}$ are mutually independent and standard Normal (i.e., zero mean and unit variance). The random variables in $\Delta\mathbf{y}$ are called *principal components* or *principal factors* by the statistics community [23].

The essence of PCA can be interpreted as a coordinate rotation of the space defined by the correlated random variables in $\Delta\mathbf{x}$. It is possible to use a small number of random variables, i.e., a small subset of principal components, to approximate the original MN-dimensional space. More details on PCA can be found in [23].

Today's statistical device models offered by most IC foundries rely on PCA to define the statistical characteristics of process variations at device level. In most statistical device model files, a set of independent standard Normal random variables are defined to capture both inter-die and intra-die variations, and the device model parameters are represented as functions of these random variables, as shown in Fig. 2. These statistical device models often accurately capture both inter-die variations and independent device mismatches. However, modeling spatially-correlated intra-die variations is not trivial and often requires a large amount of efforts for process characterization.

```
vary delta_y1  dist = gauss  std = 1
vary delta_y2  dist = gauss  std = 1

model NMOS bsim4
+ type = n
+ tox = 4e-9 + 1e-10*delta_y1  – 1.3e-10*delta_y2  + ...
+ vth0 = 0.6 + 0.24*delta_y1  + 0.3*delta_y2  + ...
+ ...
```

Fig. 2. The syntax of statistical device model is shown for a simple example where two independent standard Normal random variables delta_y1 and delta_y2 are used to model inter-die variations. Commercial statistical device models involve a large number of random variables and are substantially more complicated than this simple example.

B. Circuit-level Performance Modeling

Once the device-level variations are accurately captured by statistical device models, the following performance model can be used to analyze the variability of a given circuit performance (e.g., delay, gain, etc.)

$$f(\Delta\mathbf{y}) \approx \sum_{k=1}^{K} \alpha_k \cdot g_k(\Delta\mathbf{y}), \tag{5}$$

where f represents the circuit performance of interest, $\{\alpha_k;\ k = 1, 2, ..., K\}$ are the model coefficients, $\{g_k(\Delta\mathbf{y});\ k = 1, 2, ..., K\}$ are the basis functions (e.g., linear and quadratic polynomials), and K denotes the total number of basis functions. In other words, the circuit performance f is approximated as the linear combination of K basis functions.

The performance function $f(\Delta\mathbf{y})$ is a local perturbation around its nominal value. To approximate such a local variation, polynomial basis functions are often used [2], [4], [7]. For example, a linear performance model includes a constant term and the linear basis functions

$$\Delta y_1 \quad \Delta y_2 \quad \Delta y_3 \quad \cdots . \tag{6}$$

A simple linear performance model is sufficiently accurate, if the magnitude of process variations is relatively small. However, the linear performance model can be inaccurate, as process variations become increasingly large for today's nanoscale IC technologies. Hence, applying nonlinear basis functions to performance modeling is often required to achieve sufficiently high modeling accuracy. For instance, the

quadratic basis functions are expressed as

$$\Delta y_1^2 \quad \Delta y_1 \cdot \Delta y_2 \quad \Delta y_2^2 \quad \cdots . \tag{7}$$

A full quadratic performance model includes a constant term, the linear basis functions in (6), and the quadratic basis functions in (7).

In general, for both linear and nonlinear performance models, the unknown model coefficients $\{\alpha_k;\ k = 1, 2, ..., K\}$ in (5) can be determined by solving a set of linear equations at a number of sampling points [2], [25]

$$\sum_{k=1}^{K} \alpha_k \cdot g_k\left(\Delta \mathbf{y}^{(l)}\right) = f^{(l)} \quad \left(l = 1,2,\cdots,L\right), \tag{8}$$

where $\Delta \mathbf{y}^{(l)}$ and $f^{(l)}$ are the values of $\Delta \mathbf{y}$ and $f(\Delta \mathbf{y})$ at the l-th sampling point respectively, and L denotes the total number of sampling points. The sampling locations $\{\Delta \mathbf{y}^{(l)};\ l = 1, 2, ..., L\}$ are often determined by the method of design of experiments (DOE) [27]. Fig. 3 shows a simple DOE example where three sampling points are used to solve the coefficients for a two-dimensional linear performance model. Once the sampling locations $\{\Delta \mathbf{y}^{(l)};\ l = 1, 2, ..., L\}$ are known, the performance value $f^{(l)}$ at the l-th sampling location $\Delta \mathbf{y}^{(l)}$ is calculated by running a transistor-level simulation.

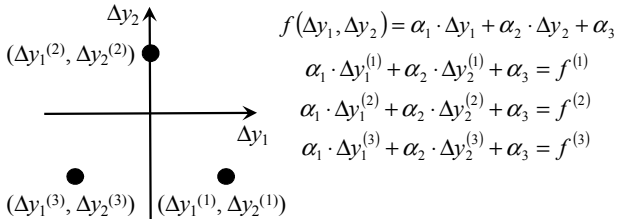

Fig. 3. A simple DOE example is shown to determine the sampling locations $(\Delta y_1^{(1)}, \Delta y_2^{(1)})$, $(\Delta y_1^{(2)}, \Delta y_2^{(2)})$ and $(\Delta y_1^{(3)}, \Delta y_2^{(3)})$ to solve the coefficients for a two-dimensional linear performance model.

To solve the linear equations in (8), we re-write them in the following matrix form

$$\mathbf{G} \cdot \boldsymbol{\alpha} = \mathbf{f}, \tag{9}$$

where

$$\mathbf{G} = \begin{bmatrix} g_1\left(\Delta \mathbf{y}^{(1)}\right) & g_2\left(\Delta \mathbf{y}^{(1)}\right) & \cdots & g_K\left(\Delta \mathbf{y}^{(1)}\right) \\ g_1\left(\Delta \mathbf{y}^{(2)}\right) & g_2\left(\Delta \mathbf{y}^{(2)}\right) & \cdots & g_K\left(\Delta \mathbf{y}^{(2)}\right) \\ \vdots & \vdots & \vdots & \vdots \\ g_1\left(\Delta \mathbf{y}^{(L)}\right) & g_2\left(\Delta \mathbf{y}^{(L)}\right) & \cdots & g_K\left(\Delta \mathbf{y}^{(L)}\right) \end{bmatrix} \tag{10}$$

$$\boldsymbol{\alpha} = \begin{bmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_K \end{bmatrix}^T \tag{11}$$

$$\mathbf{f} = \begin{bmatrix} f^{(1)} & f^{(2)} & \cdots & f^{(L)} \end{bmatrix}^T . \tag{12}$$

Many performance modeling techniques [4]-[8] assume that the number of samples (i.e., L) must be equal to or greater than the number of coefficients (i.e., K). Namely, the linear system in (9) is overdetermined. In this case, the least-squares solution of (9) can be found by solving the following optimization problem

$$\min_{\boldsymbol{\alpha}} \quad \left\| \mathbf{G} \cdot \boldsymbol{\alpha} - \mathbf{f} \right\|_2^2, \tag{13}$$

where $\|\bullet\|_2$ denotes the L_2-norm of a vector. The least-squares regression (LSR) problem in (13) can be solved by QR decomposition [24] both efficiently (i.e., with low computational cost) and robustly (i.e., with guaranteed global optimum).

The aforementioned LSR approach, however, becomes quickly intractable for large-scale performance modeling problems. In these cases, the number of unknown coefficients (i.e., K) is large (e.g., $10^4 \sim 10^6$) and, hence, LSR requires a large number of sampling points to determine all model coefficients. Remember that each sampling point is generated by running a transistor-level simulation. Hence, it can be extremely expensive, if not impossible, to repeatedly run a large number of (e.g., $10^4 \sim 10^6$) transistor-level simulations to collect all sampling points. Motivated by this observation, we will next describe a novel sparse regression (SR) method [9]-[11] that aims to create large-scale performance models (e.g., $10^4 \sim 10^6$ model coefficients) from a small set of (e.g., $10^2 \sim 10^3$) sampling points without over-fitting.

III. Sparse Regression

To minimize the required number of sampling points and, hence, the total computational cost for performance modeling, we focus on the non-trivial case where the number of samples (i.e., L) is less than the number of coefficients (i.e., K). Namely, unlike the traditional LSR that solvers model coefficients from overdetermined linear equations, we have fewer equations than unknowns and the linear system in (9) is underdetermined. In this case, the solution $\boldsymbol{\alpha}$ (i.e., the vector containing unknown model coefficients) is not unique, unless additional constraints are considered.

One possible approach of adding additional constraints to find a unique solution $\boldsymbol{\alpha}$ from the undetermined linear equations is based upon the idea of sparse regression [9]-[11]. It 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 of interest. In other words, the solution $\boldsymbol{\alpha}$ of (9) only contains a small number of non-zeros. However, we do not know the exact locations of these non-zeros. They must be identified by a "smart" algorithm based on a limited number of sampling points. In what follows, we will introduce the SR algorithm to find the unique sparse solution of the underdetermined linear system in (9).

To derive the SR method, we first show the idea of L_0-norm regularization. To this end, we formulate the following optimization problem to solve the sparse solution $\boldsymbol{\alpha}$ for (9)

$$\begin{aligned} \underset{\boldsymbol{\alpha}}{\text{minimize}} \quad & \left\| \mathbf{G} \cdot \boldsymbol{\alpha} - \mathbf{f} \right\|_2^2 \\ \text{subject to} \quad & \left\| \boldsymbol{\alpha} \right\|_0 \le \lambda \end{aligned}, \tag{14}$$

where $\|\bullet\|_0$ stands for the L_0-norm of a vector (i.e., the total number of non-zeros in the vector). The L_0-norm $\|\boldsymbol{\alpha}\|_0$ measures the sparsity of $\boldsymbol{\alpha}$. Therefore, by directly constraining the L_0-norm, the optimization in (14) attempts to find a sparse solution $\boldsymbol{\alpha}$ that minimizes the sum of squared residuals.

The parameter λ in (14) explores the tradeoff between the sparsity of the solution $\boldsymbol{\alpha}$ and the minimal value of the cost function $\|\mathbf{G} \cdot \boldsymbol{\alpha} - \mathbf{f}\|_2^2$. For instance, a large λ will result in a small cost function value, but meanwhile it will increase the

number of non-zeros in $\boldsymbol{\alpha}$. It is important to note that a small cost function value does not necessarily imply 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 the linear system in (9) is underdetermined. In the extreme case, if λ is sufficiently large and the constraint in (14) is not active, we can always find a solution $\boldsymbol{\alpha}$ 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, given a limited number of sampling points, accurately estimating the 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 [26], [28]. A Q-fold cross-validation partitions the entire data set into Q groups, as shown by the example in Fig. 4. 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., $\varepsilon = (\varepsilon_1 + \varepsilon_2 + ... + \varepsilon_Q) / Q$.

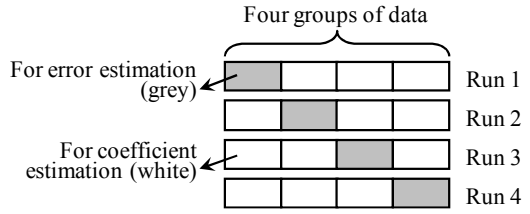

Fig. 4. A 4-fold cross-validation partitions the data set into four groups and the modeling error is estimated from four independent runs: $\varepsilon = (\varepsilon_1 + \varepsilon_2 + \varepsilon_3 + \varepsilon_4) / 4$.

For our application of SR, the optimization problem in (14) is used to repeatedly calculate the model coefficients with different λ values for multiple cross-validation runs. 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 one-dimensional function of λ. Once all cross-validation runs are complete, the final modeling error is calculated as $\varepsilon(\lambda) = [\varepsilon_1(\lambda) + \varepsilon_2(\lambda) + ... + \varepsilon_Q(\lambda)] / Q$, which is again a one-dimensional function of λ. The optimal λ is then determined by finding the minimal value of $\varepsilon(\lambda)$.

While the aforementioned L_0-norm regularization can effectively guarantee a sparse solution $\boldsymbol{\alpha}$, the optimization in (14) is NP hard and, hence, is extremely difficult to solve. An efficient technique to find the sparse solution is based on L_1-norm regularization – a relaxed version of L_0-norm [10]

$$\begin{aligned} \underset{\boldsymbol{\alpha}}{\text{minimize}} \quad & \left\| \mathbf{G} \cdot \boldsymbol{\alpha} - \mathbf{f} \right\|_2^2 \\ \text{subject to} \quad & \left\| \boldsymbol{\alpha} \right\|_1 \le \lambda \end{aligned}, \tag{15}$$

where $\|\bullet\|_1$ denotes the L_1-norm of a vector (i.e., the summation of the absolute values of all elements in the vector). The L_1-norm regularization in (15) can be re-formulated as a convex optimization problem [10] and solved both efficiently (i.e., with low computational cost) and robustly (i.e., with guaranteed global optimum).

To understand the connection between L_1-norm regularization and sparse solution, we consider a two-dimensional example (i.e., $\boldsymbol{\alpha} = [\alpha_1\ \alpha_2]^T$), as shown in Fig. 5. Since the cost function $\|\mathbf{G}\cdot\boldsymbol{\alpha} - \mathbf{f}\|_2^2$ is quadratic and positive semidefinite in $\boldsymbol{\alpha}$, its contour lines can be represented by multiple ellipsoids. On the other hand, the constraint $\|\boldsymbol{\alpha}\|_1 \le \lambda$ corresponds to a number of rotated squares, associated with different values of λ. For example, two of such squares are shown in Fig. 5 where $\lambda_1 \le \lambda_2$.

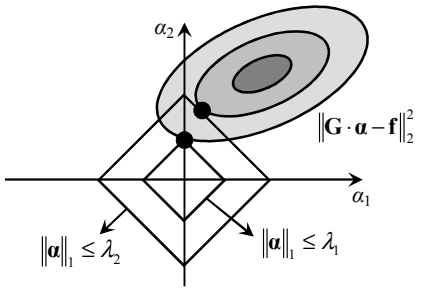

Fig. 5. L_1-norm regularization $\|\boldsymbol{\alpha}\|_1 \le \lambda$ results in a sparse solution (i.e., $\alpha_1 = 0$), if λ is sufficiently small (i.e., $\lambda = \lambda_1$).

Studying Fig. 5, 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 $\|\mathbf{G}\cdot\boldsymbol{\alpha} - \mathbf{f}\|_2^2$ eventually intersects the polytope $\|\boldsymbol{\alpha}\|_1 \le \lambda$ at one of its vertices. It, in turn, implies that one of the coefficients (i.e., α_1 in this case) becomes exactly zero. Hence, by decreasing λ of the L_1-norm regularization in (15), we can pose a strong constraint for sparsity and force a sparse solution. It, in turn, explains why L_1-norm regularization guarantees sparsity, as is the case of L_0-norm regularization.

To compare the efficacy of SR and LSR, we consider an SRAM read path example that is designed in a commercial 65 nm CMOS process [11]. The read path contains cell array, replica path for self-timing and sense amplifier, as shown in Fig. 6. In this example, both inter-die and intra-die variations are considered, and 21310 independent random variables are used to model these variations. Our objective is to model the read delay from the word line (WL) to the sense amplifier output (Out). Table I shows the linear performance modeling error and cost for both LSR and SR. Note that SR achieves about 2× error reduction over LSR in this example.

On the other hand, the overall modeling cost for both LSR and SR consists of the following two portions.

- **Simulation cost**: the cost of running a transistor-level simulator to generate all sampling points, and

- **Fitting cost**: the cost of solving all model coefficients from the sampling points.

In this example, the computational cost is dominated by transistor-level simulation, as shown in Table I. Overall, SR achieves about 25× runtime speedup over LSR. It reduces the total modeling time from 8.59 days to 8.18 hours.

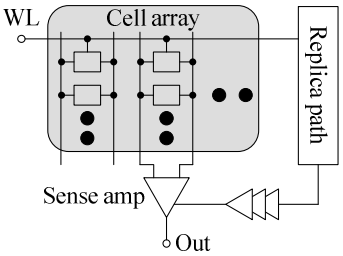

Fig. 6. A simplified circuit schematic is shown for an SRAM read path.

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

	LSR	SR
Modeling Error	9.78%	4.94%
# of Sampling Points	25000	1000
Simulation Cost (Sec.)	728250	29130
Fitting Cost (Sec.)	13856.1	338.3
Total Cost (Sec.)	742106	29468

IV. BAYESIAN MODEL FUSION

In addition to SR, Bayesian model fusion (BMF) offers another possible avenue that can be used to further reduce the modeling cost. BMF takes advantage of the fact that designing an analog and/or mixed-signal circuit typically involves multiple stages, as shown in Fig. 7(a). At each stage, the circuit is simulated to verify all performance metrics, before we move to the next design stage. In other words, simulation data are generated at each stage. Since these simulation data collected from multiple design stages come from the same circuit, they are expected to be strongly correlated. Hence, it is possible to "borrow" the simulation samples from an early design stage to further reduce the number of required simulation runs for performance modeling at a late stage.

Fig. 7(b) shows a specific application example of BMF where the schematic-level performance model is fitted by using schematic-level simulation data. Next, a model template is extracted by applying Bayesian inference to the schematic-level performance model. Finally, the model template is further calibrated by very few post-layout simulation samples to accurately create the post-layout performance model. In this application example, by "fusing" the schematic-level performance model with the post-layout performance model through a common template, the computational cost for post-layout performance modeling can be substantially reduced.

To fully understand the BMF method, we need to first explain the "common" characteristics between early-stage and late-stage performance models. To this end, we represent both the early-stage performance model $f_E(\Delta\mathbf{y})$ and the late-stage performance model $f_L(\Delta\mathbf{y})$ as the linear combinations of K basis functions, similar to (5)

$$f_E(\Delta\mathbf{y}) \approx \sum_{k=1}^{K} \alpha_{E,k} \cdot g_k(\Delta\mathbf{y}) \tag{16}$$

$$f_L(\Delta\mathbf{y}) \approx \sum_{k=1}^{K} \alpha_{L,k} \cdot g_k(\Delta\mathbf{y}), \tag{17}$$

where $\{\alpha_{E,k};\ k = 1, 2, ..., K\}$ and $\{\alpha_{L,k};\ k = 1, 2, ..., K\}$ denote the early-stage and late-stage model coefficients respectively. Since both $f_E(\Delta\mathbf{y})$ and $f_L(\Delta\mathbf{y})$ model the same performance metric of the same circuit, we expect that the magnitude of the model coefficients $\alpha_{E,k}$ and $\alpha_{L,k}$ is similar. Such "prior" knowledge can be statistically encoded as

$$\alpha_{L,k} \sim N\left(0, \alpha_{E,k}^2\right). \tag{18}$$

Eq. (18) means that each late-stage model coefficient $\alpha_{L,k}$ is assumed to follow a zero-mean Normal distribution as a prior. The variance of the Normal distribution equals $\alpha_{E,k}^2$ that is determined by the magnitude of the early-stage model coefficient $\alpha_{E,k}$. In other words, if the magnitude of the early-stage coefficient $\alpha_{E,k}$ is large, the magnitude of the late-stage coefficient $\alpha_{L,k}$ is also "likely" to be large.

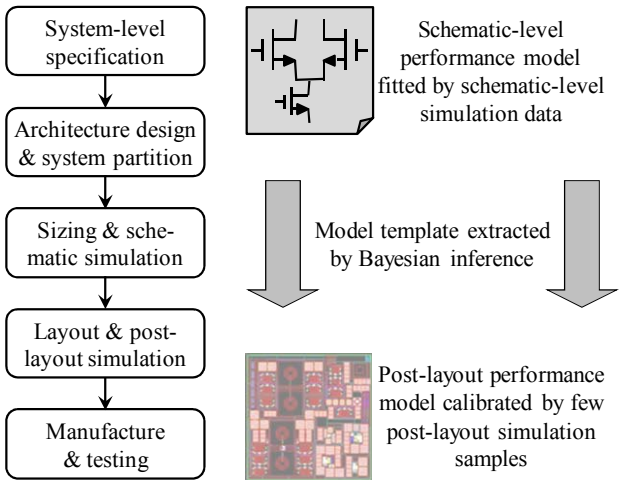

Fig. 7. (a) Designing an analog and/or mixed-signal circuit typically involves multiple stages. (b) A specific application example of Bayesian model fusion (BMF) is shown to borrow schematic-level simulation data to efficiently create post-layout performance model.

In addition to the aforementioned "similarity" between model coefficients, BMF further assumes that the error of both early-stage and late-stage performance models is similar. Here, the modeling error associated with the l-th sampling point is modeled as a zero-mean Normal distribution

$$e_l \sim N\left(0, \sigma^2\right), \tag{20}$$

where σ^2 denotes the standard deviation of the Normal distribution and it controls the magnitude of the modeling error. Once the early-stage performance model $f_E(\Delta\mathbf{y})$ is fitted from L_E early-stage sampling points, the value of σ can be estimated as

$$\sigma = \sqrt{\frac{1}{L_E} \cdot \sum_{l=1}^{L_E} \left[\sum_{k=1}^{K} \alpha_{E,k} \cdot g_k\left(\Delta\mathbf{y}^{(l)}\right) - f_E^{(l)} \right]^2}, \tag{20}$$

where $\Delta\mathbf{y}^{(l)}$ and $f_E^{(l)}$ are the values of $\Delta\mathbf{y}$ and $f_E(\Delta\mathbf{y})$ at the l-th sampling point respectively.

By statistically defining the similarity between early-stage and late-stage performance models, we are now ready to apply a Bayesian inference to accurately estimate the late-stage model coefficients $\{\alpha_{L,k};\ k = 1, 2, ..., K\}$ with a small number of late-stage simulation runs. Fig. 8 summarizes the overall flow of the aforementioned BMF method. The mathematical details of BMF are not included in this paper due to the constraint on page limit.

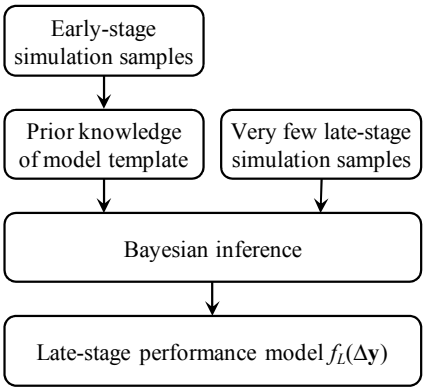

Fig. 8. A simplified flow of the Bayesian model fusion (BMF) method is summarized for late-stage performance modeling with a small number of late-stage simulation samples.

To compare the efficiency between BMF and SR, we consider an SRAM read path example that is designed in a 32 nm CMOS process. The topology of the read path is similar to what is shown in Fig. 6. The read delay from the word line (WL) to the sense amplifier output (Out) is the performance of interest. Our objective is to efficiently determine the post-layout delay model by re-using the schematic-level simulation data.

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

	SR	BMF
Modeling Error	8.14%	5.88%
# of Post-Layout Sampling Points	800	100
Simulation Cost (Sec.)	147656	18457
Fitting Cost (Sec.)	695	15
Total Cost (Sec.)	148351	18472

In this example, there are 47937 independent random variables that are used to model both inter-die and intra-die variations. Compared to the 65 nm SRAM read path shown in Section III, the total number of independent random variables further increases, as we move from 65 nm to 32 nm. Table II shows the linear performance modeling error and cost for both SR and BMF. Note that BMF achieves about 8× runtime speedup over SR, while providing superior modeling accuracy in this example. It reduces the total modeling time from 1.72 days to 5.13 hours.

V. PERFORMANCE MODELING APPLICATIONS

In this section, we briefly discuss three important applications of performance modeling, including (i) critical variation source identification, (ii) parametric yield estimation and optimization, and (iii) application-specific worst-case corner extraction. In what follows, we will discuss each of these applications and highlight their practical utility.

A. Critical Variation Source Identification

The performance model $f(\Delta\mathbf{y})$ in (5) captures the relation between the circuit-level performance metric f and the device-level variation sources $\{\Delta y_i;\ i = 1, 2, ..., MN\}$. Hence, once $f(\Delta\mathbf{y})$ is determined, it provides the useful information that helps to quantitatively assess the impact of each device-level variation source for the given performance of interest.

Considering the linear performance model as an example, we re-write (5) by using the linear basis functions and a constant term

$$f(\Delta\mathbf{y}) \approx \alpha_1 \cdot \Delta y_1 + \alpha_2 \cdot \Delta y_2 + \cdots + \alpha_{MN} \cdot \Delta y_{MN} + \alpha_{MN+1} . \quad (21)$$

In (21), the model coefficient α_i provides the sensitivity information for the performance f with respect to each device-level variation source Δy_i. In other words, if the magnitude of α_i is large, the device-level variation source Δy_i is critical as it substantially contributes to the variability of the circuit-level performance f.

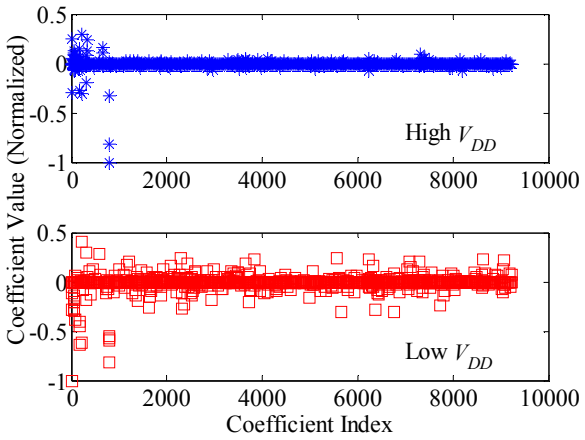

Fig. 9. Linear performance model coefficients (normalized) are shown for the phase noise of a voltage-controlled oscillator (VCO) that operates at two different modes: high V_{DD} and low V_{DD}. Each coefficient provides the sensitivity information for the phase noise with respect to the corresponding device-level variation source.

As an example for illustration purpose, we consider a voltage-controlled oscillator (VCO) example that is designed in a commercial 32 nm CMOS process. In this example, there are 9231 independent random variables that are used to model both inter-die and intra-die variations. Our objective is to model the VCO phase noise as a linear function of all device-level variation sources. Two different phase noise models are fitted, corresponding to two different operation modes: high V_{DD} and low V_{DD}. Fig. 9 plots the coefficient values for both phase noise models. The magnitude of these model coefficients helps us to identify the critical variation sources that substantially impact the phase noise in this example. It is interesting to note that a large number of device-level variation sources become critical in the low V_{DD} mode. This observation is consistent with our design intuition. Namely, as the supply voltage decreases, the VCO circuit becomes increasingly sensitive to process variations.

B. *Parametric Yield Estimation and Optimization*

Another important application of statistical performance modeling is for efficient yield estimation and optimization. For example, given the performance model $f(\Delta\mathbf{y})$ in (5), a novel algorithm of asymptotic probability extraction (APEX) has been proposed in [12] to efficiently estimate the probability distribution of the performance f. Based on the analytical function $f(\Delta\mathbf{y})$, APEX first calculates the high-order moments of f. Next, it finds the probability density function $pdf(f)$ such that the high-order moments of $pdf(f)$ match those of $f(\Delta\mathbf{y})$. Once the performance distribution $pdf(f)$ is known, the parametric yield associated with the performance metric f can be easily calculated from its distribution. Such a moment-matching approach has been further extended to estimate the parametric yield defined by multiple correlated analog/mixed-signal performance metrics in [13].

In addition to parametric yield estimation, statistical performance modeling can also be applied to guide parametric yield optimization. In particular, performance modeling is often used to facilitate efficient yield optimization through two different avenues [14]-[18]. First, performance models can be fitted during each iteration step within an optimization loop to efficiently assess the quality of a given circuit design (i.e., estimate its parametric yield). Second, performance models can be created to capture the relation between the parametric yield (i.e., the merit function that we aim to maximize) and the design variables (i.e., the optimization variables that we need to determine). Such a yield model provides valuable information to assist the optimization engine to quickly explore the design space.

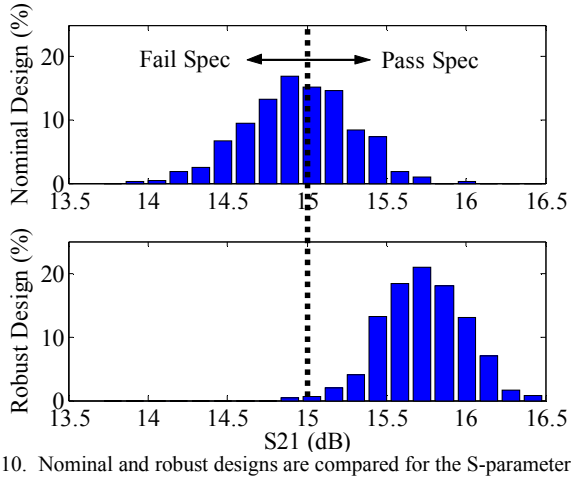

Fig. 10. Nominal and robust designs are compared for the S-parameter S21 of a low noise amplifier (LNA) circuit. The robust design is optimized for parametric yield and is able to leave sufficient margin to accommodate process variations.

To intuitively explain the difference between the aforementioned parametric yield optimization and the traditional nominal design optimization, we consider a low noise amplifier (LNA) example that is designed in a commercial 0.25 μm BiCMOS process. In this example, the design variables include the lengths and widths of all transistors, the geometrical sizes of all passive devices such as capacitors and inductors, and the bias current value. The performance metrics include center frequency, S-parameters, noise figure, nonlinear distortion, and power consumption. The detailed setup of this LNA example can be found in [18].

Our objective here is to determine the optimal values of all design variables such that the power consumption of the LNA is minimized while the parametric yield is sufficiently high. Fig. 10 shows the probability distribution of S21 (i.e., one of the S-parameters) for both the nominal design (i.e., optimized without considering process variations) and the robust design (i.e., optimized with consideration of parametric yield). Studying Fig. 10 reveals an important fact that the nominal design pushes the S-parameter S21 to the boundary of its specification. Therefore, the worst-case S21 violates the design requirement and the parametric yield of the nominal design is less than 50%. On the other hand, the robust design is able to leave sufficient margin for the performance metric S21. The additional performance margin enables the LNA circuit to meet the given specification with high parametric yield.

C. *Application-Specific Worst-Case Corner Extraction*

The last application discussed by this paper is to extract application-specific worst-case corners based on statistical performance models. Namely, we aim to find the unique process corner at which a given circuit fails to work. Once the worst-case corners are determined, analog/mixed-signal circuit designers can simulate their circuit at these corners, find the reason for its performance failure, and eventually come up with the appropriate solution to fix the problem. Hence, worst-case corner extraction enables circuit designers to easily debug the circuit so that they can efficiently use their design knowledge to improve parametric yield.

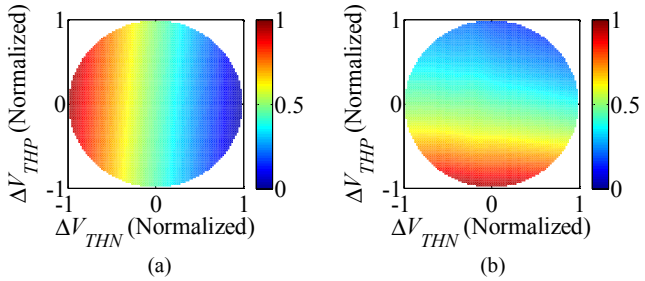

Fig. 11. The performance variations (normalized) of a voltage-controlled oscillator (VCO) are shown as functions of two random variables ΔV_{THP} and ΔV_{THN} for: (a) the center frequency F_0, and (b) the gain K_{VCO}.

Accurately identifying the worst-case corners for analog and mixed-signal circuits, however, is not trivial. Unlike digital circuits for which the process corners (e.g., the FF and SS corners) are often defined by IC foundries, the worst-case corners for analog and mixed-signal circuits are topology-dependent and performance-dependent. To illustrate this "application-specific" characteristic of analog/mixed-signal worst-case corners, we consider a voltage-controlled oscillator (VCO) example that is designed in a commercial 0.13 μm CMOS process. For visualization purpose, we only consider two random variables ΔV_{THP} and ΔV_{THN} that model the inter-die variations of threshold voltage for PMOS and NMOS

transistors respectively. Fig. 11 shows how the center frequency F_0 and the gain K_{VCO} vary due to the process variations of ΔV_{THP} and ΔV_{THN}. Note that the variation patterns of F_0 and K_{VCO} are substantially different. Since the gradient directions of F_0 and K_{VCO} with respect to process variations are completely different, the worst-case corners for F_0 and K_{VCO} are expected to be different. Hence, an efficient algorithm must be used to find the "application-specific" worst-case corners for these two different performance metrics respectively.

Several algorithms have been developed in the literature to address the aforementioned worst-case corner extraction problem [19]-[20]. Given the performance model $f(\Delta\mathbf{y})$ in (5), these corner extraction methods find the unique process corner at which the circuit performance f reaches its worst-case value (e.g., the 99% point of its cumulative distribution function). Such a mathematical definition of worst-case corner can be translated to a convex optimization problem and solved both efficiently (i.e., with low computational cost) and robustly (i.e., with guaranteed global optimum). More details about application-specific worst-case corner extraction can be found in [19]-[20].

VI. Conclusions

Designing analog and mixed-signal circuits has become increasingly challenging with today's nanoscale IC technologies. Large-scale process variations must be appropriately modeled, analyzed and optimized to ensure high parametric yield. In this paper, we discuss the recent progress of statistical performance modeling (i.e., sparse regression and Bayesian model fusion) and its important applications (i.e., critical variation source identification, parametric yield estimation and optimization, and application-specific worst-case corner extraction). The efficacy of these methods is compared to other traditional approaches by using several analog and mixed-signal circuit examples.

VII. Acknowledgement

The authors acknowledge the support of the C2S2 Focus Center, one of six research centers funded under the Focus Center Research Program (FCRP), a Semiconductor Research Corporation entity. This work is also supported in part by the National Science Foundation under contract CCF–0811023.

VIII. References

[1] S. Nassif, "Modeling and analysis of manufacturing variations," *IEEE CICC*, pp. 223-228, 2001.

[2] X. Li, J. Le and L. Pileggi, *Statistical Performance Modeling and Optimization*, Now Publishers, 2007.

[3] Semiconductor Industry Associate, *International Technology Roadmap for Semiconductors*, 2011.

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

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

[6] 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.

[7] 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.

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

[9] 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.

[10] X. Li, "Finding deterministic solution from underdetermined equation: large-scale performance modeling by least angle regression," *IEEE DAC*, pp. 364-369, 2009.

[11] X. Li, "Finding deterministic solution from underdetermined equation: large-scale performance modeling of analog/RF circuits," *IEEE Trans. CAD*, vol. 29, no. 11, pp. 1661-1668, Nov. 2010.

[12] X. Li, J. Le, P. Gopalakrishnan and L. Pileggi, "Asymptotic probability extraction for nonnormal performance distributions," *IEEE Trans. CAD*, vol. 26, no. 1, pp. 16-37, Jan. 2007.

[13] X. Li, Y. Zhang and L. Pileggi, "Quadratic statistical MAX approximation for parametric yield estimation of analog/RF integrated circuits," *IEEE Trans. on CAD*, vol. 27, no. 5, pp. 831-843, May. 2008.

[14] Z. Wang and S. Director, "An efficient yield optimization method using a two step linear approximation of circuit performance," *IEEE EDAC*, pp. 567-571, 1994.

[15] A. Dharchoudhury and S. Kang, "Worse-case analysis and optimization of VLSI circuit performance," *IEEE Trans. CAD*, vol. 14, no. 4, pp. 481-492, Apr. 1995.

[16] G. Debyser and G. Gielen, "Efficient analog circuit synthesis with simultaneous yield and robustness optimization," *IEEE ICCAD*, pp. 308-311, 1998.

[17] 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," *IEEE DAC*, pp. 858-863, 2001.

[18] X. Li, P. Gopalakrishnan, Y. Xu and L. Pileggi, "Robust analog/RF circuit design with projection-based performance modeling," *IEEE Trans. on CAD*, vol. 26, no. 1, pp. 2-15, Jan. 2007.

[19] M. Sengupta, S. Saxena, L. Daldoss, G. Kramer, S. Minehane and J. Cheng, "Application-specific worst case corners using response surfaces and statistical models," *IEEE Trans. CAD*, vol. 24, no. 9, pp. 1372-1380, 2005.

[20] H. Zhang, T. Chen, M. Ting and X. Li, "Efficient design-specific worst-case corner extraction for integrated circuits," *IEEE DAC*, pp. 386-389, 2009.

[21] B. Stine, D. Boning and J. Chung, "Analysis and decomposition of spatial variation in integrated circuit processes and devices," *IEEE Trans. Semiconductor Manufacturing*, vol. 10, no. 1, pp. 24-41, Feb. 1997.

[22] J. Pelgrom, C. Duinmaijer and P. Welbers, "Matching properties of MOS transistors", *IEEE JSSC*, vol. 25, no. 5, pp. 1433-1439, Oct. 1989.

[23] G. Seber, *Multivariate Observations*, John Wiley & Sons, 1984.

[24] G. Golub and C. Loan, *Matrix Computations*, The Johns Hopkins Univ. Press, 1996.

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

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

[27] D. Montgomery, *Design and Analysis of Experiments*, John Wiley & Sons, 2005.

[28] C. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.