Bayesian Model Fusion: Large-Scale Performance Modeling of Analog and Mixed-Signal Circuits by Reusing Early-Stage Data

Fa Wang[1], Wangyang Zhang[1], Shupeng Sun[1], Xin Li[1] and Chenjie Gu[2]
[1]ECE Department, Carnegie Mellon University, Pittsburgh, PA 15213
[2]Strategic CAD Labs, Intel Corporation, Hillsboro, OR 97124
{fwang1, wyzhang, shupengs, xinli}@ece.cmu.edu, chenjie.gu@intel.com

ABSTRACT

Efficient high-dimensional performance modeling of today's complex analog and mixed-signal (AMS) circuits with large-scale process variations is an important yet challenging task. In this paper, we propose a novel performance modeling algorithm that is referred to as Bayesian Model Fusion (BMF). Our key idea is to borrow the simulation data generated from an early stage (e.g., schematic level) to facilitate efficient high-dimensional performance modeling at a late stage (e.g., post layout) with low computational cost. Such a goal is achieved by statistically modeling the performance correlation between early and late stages through Bayesian inference. Several circuit examples designed in a commercial 32nm CMOS process demonstrate that BMF achieves up to 9× runtime speedup over the traditional modeling technique without surrendering any accuracy.

1. INTRODUCTION

The aggressive scaling of integrated circuits (ICs) leads to large-scale process variations that cannot be easily reduced by foundries. Process variations manifest themselves as the uncertainties associated with the geometrical and electrical parameters of semiconductor devices. These device-level variations significantly impact the parametric yield of analog and mixed-signal (AMS) circuits and, hence, must be appropriately modeled, analyzed and optimized at all levels of design hierarchy [1]-[2].

To address this variability issue, various techniques for performance modeling have been developed during the past two decades [3]-[8]. The objective is to approximate the circuit-level performance (e.g., gain of an analog amplifier) as an analytical (e.g., linear, quadratic, etc) function of device-level variations (e.g., ΔV_{TH}, ΔT_{OX}, etc). Once such a performance model is available, it can be applied to a number of important applications such as estimating parametric yield [9], extracting worst-case corner [10], optimizing circuit design [11]-[15], etc.

While performance modeling was extensively studied in the past, the evolution of today's AMS circuits has posed a number of new challenges in this area. In particular, the recent adoption of several emerging design methodologies (e.g., reconfigurable analog design, adaptive post-silicon tuning, etc) leads to highly complex AMS systems that integrate numerous nanoscale devices. The remarkable increase of AMS circuit size results in a two-fold consequence.

DAC'13, May 29 - June 07 2013, Austin, TX, USA.

- *High-dimensional variation space*: A large number of device-level random variables must be used to model the process variations associated with a large-scale AMS system. For example, about 40 independent random variables are required to model the device mismatches of a single transistor for a commercial 32nm CMOS process. If an AMS system contains 10^4 transistors, there are about 4×10^5 random variables in total to capture the corresponding device-level variations, resulting in a high-dimensional variation space. In addition, it is extremely difficult, if not impossible, to pre-select a subset of these random variables for variation analysis, since the impact of device mismatches is circuit- and performance-dependent.
- *Expensive circuit simulation*: The computational cost of circuit simulation substantially increases, as the AMS circuit size becomes increasingly large. For instance, it may take a few days or even a few weeks to run the transistor-level simulation of a large AMS circuit such as phase-locked loop or high-speed link.

These recent trends of today's AMS circuits make performance modeling extremely difficult. On one hand, a large number of simulation samples must be generated in order to fit a high-dimensional model. On the other hand, creating a single sampling point by transistor-level simulation can take a large amount of computational time. The challenging issue here is how to make performance modeling computationally *affordable* for today's large-scale AMS circuits. This fundamental issue has not been appropriately addressed by the traditional performance modeling techniques, e.g., the recent sparse regression algorithm based on Orthogonal Matching Pursuit (OMP) [8].

In this paper, we propose a new *Bayesian Model Fusion* (BMF) technique to facilitate large-scale performance modeling of AMS circuits. The proposed BMF method is motivated by the fact that today's AMS circuits are often designed via a multi-stage flow. Namely, an AMS design often spans three core stages: (i) schematic design, (ii) layout design, and (iii) chip manufacturing and testing. At each stage, simulation or measurement data are collected to validate the circuit design, before moving to the next stage. The traditional performance modeling techniques rely on the data at a single stage only and they completely ignore the data that are generated at other stages. The key idea of BMF, however, is to *reuse* the early-stage data when fitting a late-stage performance model. As such, the performance modeling cost can be substantially reduced.

Mathematically, the proposed BMF method is derived from the theory of Bayesian inference. Starting from a set of early-stage (e.g., schematic-level) sampling points, BMF first approximates an early-stage performance model based on these samples. The early-stage model is used as a template to define our prior knowledge for late-stage (e.g., post-layout) performance modeling. Specifically, a prior distribution is statistically defined for the late-stage model coefficients. The prior knowledge is then combined with very few late-stage sampling points to solve the

late-stage model coefficients via Bayesian inference. From this point of view, by *fusing* the early-stage and late-stage performance models through Bayesian inference, we only need a small number of late-stage sampling points to fit a high-dimensional late-stage model, thereby significantly reducing the computational cost for performance modeling. As will be demonstrated by our numerical examples in Section 5, BMF achieves up to 9× runtime speedup over the traditional modeling technique without surrendering any accuracy.

BMF was previously proposed for parametric yield estimation of AMS circuits in [16] where Bayesian inference was used to estimate the probability distribution of AMS performance metrics. In this paper, we further extend the idea of BMF to performance modeling. It is important to emphasize that the formulation of our prior knowledge for performance modeling is completely different from that shown in [16], as will be discussed in Section 3.

The remainder of this paper is organized as follows. In Section 2, we review the important background on performance modeling, and then derive our proposed BMF method in Section 3. Several implementation issues are discussed in Section 4. The efficacy of BMF is demonstrated by several circuit examples in Section 5. Finally, we conclude in Section 6.

2. BACKGROUND

Given an AMS circuit (e.g., an analog amplifier), its performance (e.g., gain) may vary due to device-level variations (e.g., ΔV_{TH}, ΔT_{OX}, etc). The objective of performance modeling is to approximate the circuit performance as an analytical function of the device-level variations:

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

where f represents the performance of interest, $\mathbf{x}$ is a vector containing the random variables to model device-level variations, $\{\alpha_m;\ m = 1, 2, ..., M\}$ denote the model coefficients, $\{g_m(\mathbf{x});\ m = 1, 2, ..., M\}$ are the basis functions (e.g., linear or quadratic polynomials), and M is the total number of basis functions.

In order to determine the performance model in (1), we need to find the model coefficients $\{\alpha_m;\ m = 1, 2, ..., M\}$. Towards this goal, the traditional least-squares fitting method first generates a set of sampling points and then solves the model coefficients from the following linear equation [18]:

$$\mathbf{G} \cdot \boldsymbol{\alpha} = \mathbf{f} \tag{2}$$

where

$$\mathbf{G} = \begin{bmatrix} g_1(\mathbf{x}^{(1)}) & g_2(\mathbf{x}^{(1)}) & \cdots & g_M(\mathbf{x}^{(1)}) \\ g_1(\mathbf{x}^{(2)}) & g_2(\mathbf{x}^{(2)}) & \cdots & g_M(\mathbf{x}^{(2)}) \\ \vdots & \vdots & \vdots & \vdots \\ g_1(\mathbf{x}^{(K)}) & g_2(\mathbf{x}^{(K)}) & & g_M(\mathbf{x}^{(K)}) \end{bmatrix} \tag{3}$$

$$\boldsymbol{\alpha} = \begin{bmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_M \end{bmatrix}^T \tag{4}$$

$$\mathbf{f} = \begin{bmatrix} f^{(1)} & f^{(2)} & \cdots & f^{(K)} \end{bmatrix}^T. \tag{5}$$

In (3)-(5), $\mathbf{x}^{(k)}$ and $f^{(k)}$ are the values of $\mathbf{x}$ and $f(\mathbf{x})$ at the kth sampling point respectively, and K represents the total number of sampling points. The number of sampling points (i.e., K) should be greater than the number of unknown coefficients (i.e., M). As such, the linear equation in (2) is overdetermined and the unknown model coefficients $\{\alpha_m;\ m = 1, 2, ..., M\}$ are found by solving its least-squares solution.

When the aforementioned least-squares fitting method is applied to fit a high-dimensional performance model with many unknown model coefficients, it requires a large number of sampling points to form the overdetermined linear equation in (2). Note that each sampling point is generated by running an expensive transistor-level simulation. It, in turn, implies that the least-squares fitting approach can be extremely expensive for high-dimensional performance modeling.

Recently, sparse regression has been developed to address this complexity issue [8]. The key idea is not to solve an overdetermined linear equation. Instead, the unknown model coefficients are uniquely determined by solving an underdetermined linear equation. This goal is achieved by exploiting the fact that most model coefficients of a high-dimensional performance model are close to zero. In other words, the unknown model coefficients carry a unique *sparse* pattern. The sparse regression algorithms were particularly developed to solve these sparse coefficients from a small number of sampling points. As such, the simulation cost of generating the required sampling points is greatly reduced.

While sparse regression has been successfully applied to many practical applications, it still requires a large number of (e.g., 10^3) sampling points to fit a high-dimensional performance model [8]. Therefore, it remains ill-equipped for modeling large-scale AMS circuits where running a single transistor-level simulation to generate one sampling point may take a few days or even a few weeks. Motivated by this observation, we will propose a new BMF technique in this paper to further reduce the number of required simulation samples and, hence, the computational cost for large-scale performance modeling.

3. BAYESIAN MODEL FUSION

Similar to sparse regression, the proposed BMF method relies on the assumption that most model coefficients of a high-dimensional performance model are close to zero. However, unlike the traditional sparse regression approach that fits the sparse performance model based on the simulation data at a single stage only (e.g., post-layout simulation data), BMF attempts to identify the underlying sparse pattern by re-using the early-stage data (e.g., schematic-level simulation data) in order to efficiently fit a late-stage (e.g., post-layout) performance model. In particular, BMF consists of the following two major steps: (i) statistically defining the prior knowledge of the sparse pattern based on the early-stage simulation data, and (ii) optimally determining the late-stage performance model by combining the prior knowledge and very few late-stage simulation samples. In this section, we will discuss the mathematical formulation of these two steps and highlight the novelty.

3.1 Prior Knowledge Definition

We consider two different performance models: the early-stage model $f_E(\mathbf{x})$ and the late-stage model $f_L(\mathbf{x})$:

$$f_E(\mathbf{x}) \approx \sum_{m=1}^{M} \alpha_{E,m} \cdot g_m(\mathbf{x}) \tag{6}$$

$$f_L(\mathbf{x}) \approx \sum_{m=1}^{M} \alpha_{L,m} \cdot g_m(\mathbf{x}) \tag{7}$$

where $\{\alpha_{E,m};\ m = 1, 2, ..., M\}$ and $\{\alpha_{L,m};\ m = 1, 2, ..., M\}$ represent the early-stage and late-stage model coefficients, respectively. In (6)-(7), we assume that the early-stage model $f_E(\mathbf{x})$ and the late-stage model $f_L(\mathbf{x})$ share the same basis functions. More complicated cases where $f_E(\mathbf{x})$ and $f_L(\mathbf{x})$ are approximated by different basis functions will be further discussed in Section 4.1.

The early-stage model $f_E(\mathbf{x})$ is fitted from the early-stage simulation data. In practice, the early-stage simulation data are collected to validate the early-stage design, before we move to the next stage. For this reason, we should already know the early-stage model $f_E(\mathbf{x})$ before fitting the late-stage model $f_L(\mathbf{x})$. Namely, we assume that the early-stage model coefficients $\{\alpha_{E,m}; m = 1, 2, ..., M\}$ are provided as the input to our proposed BMF method for late-stage performance modeling.

Given the early-stage model $f_E(\mathbf{x})$, we first extract the prior knowledge that can be used to facilitate efficient late-stage modeling. To this end, we propose to learn the underlying sparse pattern for the late-stage model $f_L(\mathbf{x})$ based on the early-stage model coefficients $\{\alpha_{E,m}; m = 1, 2, ..., M\}$. Remember that both the early-stage and late-stage models are fitted for the same performance metric of the same circuit. Their model coefficients should be similar. Namely, if the early-stage model coefficient $\alpha_{E,m}$ has a large (or small) magnitude, it is likely that the late-stage model coefficient $\alpha_{L,m}$ also has a large (or small) magnitude. Such prior knowledge should be mathematically encoded into our proposed performance modeling flow.

In this paper, we statistically represent the prior knowledge as a probability density function (PDF) that is referred to as the *prior distribution* [19]. In particular, we model each late-stage model coefficient as a zero-mean Gaussian distribution:

$$pdf\left(\alpha_{L,m}\right)=\frac{1}{\sqrt{2\pi}\cdot\sigma_m}\cdot\exp\left(-\frac{\alpha_{L,m}^2}{2\cdot\sigma_m^2}\right)\sim N\left(0,\sigma_m^2\right)\quad\left(m=1,2,\cdots,M\right) \tag{8}$$

where the standard deviation σ_m is a parameter that encodes the magnitude information of the model coefficient $\alpha_{L,m}$. If the standard deviation σ_m is small, the prior distribution $pdf(\alpha_{L,m})$ is narrowly peaked around zero, implying that the coefficient $\alpha_{L,m}$ is possibly close to zero. Otherwise, if the standard deviation σ_m is large, the prior distribution $pdf(\alpha_{L,m})$ widely spreads over a large range and the coefficient $\alpha_{L,m}$ can possibly take a value that is far away from zero. Figure 1 shows a simple example of our proposed prior distribution for two model coefficients $\alpha_{L,1}$ and $\alpha_{L,2}$ where σ_1 is small and σ_2 is large.

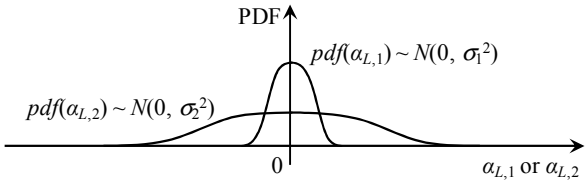

Figure 1. A simple example of our proposed prior distribution is shown for two model coefficients $\alpha_{L,1}$ and $\alpha_{L,2}$. The coefficient $\alpha_{L,1}$ is possibly close to zero, since its prior distribution is narrowly peaked around zero. The coefficient $\alpha_{L,2}$ can possibly be far away from zero, since its prior distribution widely spreads over a large range.

Given (8), we need to appropriately determine the standard deviation σ_m to fully specify the prior distribution $pdf(\alpha_{L,m})$. The value of σ_m should be optimized so that the probability distribution $pdf(\alpha_{L,m})$ correctly represents our prior knowledge. In other words, by appropriately choosing the value of σ_m, the prior distribution $pdf(\alpha_{L,m})$ should take a large value (i.e., a high probability) at the location where the actual late-stage model coefficient $\alpha_{L,m}$ occurs. However, we only know the early-stage model coefficient $\alpha_{E,m}$, instead of the late-stage model coefficient $\alpha_{L,m}$, at this moment. Remember that $\alpha_{E,m}$ and $\alpha_{L,m}$ are expected to be similar. Hence, the prior distribution $pdf(\alpha_{L,m})$ should also take a large value at $\alpha_{L,m} = \alpha_{E,m}$. Based on this criterion, the optimal prior distribution $pdf(\alpha_{L,m})$ can be found by maximizing the probability for $\alpha_{E,m}$ to occur:

$$\max_{\sigma_m}\quad pdf\left(\alpha_{L,m}=\alpha_{E,m}\right)\quad\left(m=1,2,\cdots,M\right). \tag{9}$$

Namely, given the early-stage model coefficient $\alpha_{E,m}$, the optimal standard deviation σ_m is determined by the *maximum likelihood estimation* (MLE) in (9).

To solve σ_m from (9), we consider the following first-order optimality condition:

$$\frac{d}{d\sigma_m}pdf\left(\alpha_{L,m}=\alpha_{E,m}\right)=0\quad\left(m=1,2,\cdots,M\right). \tag{10}$$

Substituting (8) into (10) yields:

$$\frac{1}{\sqrt{2\pi}\cdot\sigma_m}\cdot\exp\left(-\frac{\alpha_{E,m}^2}{2\cdot\sigma_m^2}\right)\cdot\left(\frac{\alpha_{E,m}^2}{\sigma_m^3}-\frac{1}{\sigma_m}\right)=0\quad\left(m=1,2,\cdots,M\right). \tag{11}$$

The optimal value of σ_m is equal to:

$$\sigma_m=\left|\alpha_{E,m}\right|\quad\left(m=1,2,\cdots,M\right). \tag{12}$$

Eq. (12) reveals an important fact that the optimal standard deviation σ_m is simply equal to the absolute value of the early-stage model coefficient $|\alpha_{E,m}|$. This observation is consistent with our intuition. Namely, if the early-stage model coefficient $\alpha_{E,m}$ has a large (or small) magnitude, the late-stage model coefficient $\alpha_{L,m}$ should also have a large (or small) magnitude and, hence, the standard deviation σ_m should be large (or small).

To complete the definition of the prior distribution for all late-stage model coefficients $\{\alpha_{L,m}; m = 1, 2, ..., M\}$, we further assume that these coefficients are statistically independent and their joint distribution is represented as:

$$pdf\left(\boldsymbol{\alpha}_L\right)=\frac{1}{\left(\sqrt{2\pi}\right)^M\cdot\prod_{m=1}^{M}\sigma_m}\cdot\exp\left(-\sum_{m=1}^{M}\frac{\alpha_{L,m}^2}{2\cdot\sigma_m^2}\right) \tag{13}$$

where

$$\boldsymbol{\alpha}_L=\left[\alpha_{L,1}\quad\alpha_{L,2}\quad\cdots\quad\alpha_{L,M}\right]^T \tag{14}$$

contains all late-stage model coefficients. The independence assumption in (13) simply implies that we do not know the correlation information among these coefficients as our prior knowledge. The correlation information will be learned from the late-stage simulation data, when the posterior distribution is calculated by the Bayesian inference in Section 3.2.

Finally, it is important to mention that the prior knowledge can be possibly defined as a distribution that is different from (8). For example, the prior distribution is specified as a Gaussian distribution with non-zero mean in [16]. It, however, does not encode the sparse pattern of model coefficients, as is the case of this paper. The efficacy of different prior definitions is case-dependent. It remains an open question how to determine the optimal prior distribution for a specific performance modeling problem where the circuit and performance of interest are given. This problem will be further studied in our future research.

3.2 Maximum-A-Posteriori Estimation

Once the prior distribution $pdf(\boldsymbol{\alpha}_L)$ is derived in (13), we will combine $pdf(\boldsymbol{\alpha}_L)$ with K late-stage simulation samples $\{(\mathbf{x}^{(k)}, f_L^{(k)}); k = 1, 2, \ldots, K\}$, where $\mathbf{x}^{(k)}$ and $f_L^{(k)}$ are the values of $\mathbf{x}$ and $f_L(\mathbf{x})$ at

the kth sampling point respectively, to solve the late-stage model coefficient $\boldsymbol{\alpha}_L$ by *maximum-a-posteriori* (MAP) estimation. The key idea of MAP is to find the *posterior distribution* [19], i.e., the conditional PDF $pdf(\boldsymbol{\alpha}_L \mid \mathbf{f}_L)$ where

$$\mathbf{f}_L = \left[f_L^{(1)} \quad f_L^{(2)} \quad \cdots \quad f_L^{(K)} \right]^T \tag{15}$$

contains all late-stage simulation samples that are collected. Intuitively, the posterior distribution $pdf(\boldsymbol{\alpha}_L \mid \mathbf{f}_L)$ indicates the remaining uncertainty of $\boldsymbol{\alpha}_L$, after we observe K late-stage simulation samples. Here, since $\boldsymbol{\alpha}_L$ is a random variable, it is described by a probability distribution, instead of a deterministic value. MAP attempts to find the optimal value of $\boldsymbol{\alpha}_L$ to maximize the posterior distribution $pdf(\boldsymbol{\alpha}_L \mid \mathbf{f}_L)$. Namely, it aims to find the solution $\boldsymbol{\alpha}_L$ that is most likely to occur according to the posterior distribution.

Based on Bayes' theorem, the posterior distribution $pdf(\boldsymbol{\alpha}_L \mid \mathbf{f}_L)$ is proportional to the prior distribution $pdf(\boldsymbol{\alpha}_L)$ multiplied by the likelihood function $pdf(\mathbf{f}_L \mid \boldsymbol{\alpha}_L)$ [19]:

$$pdf\left(\boldsymbol{\alpha}_L \middle| \mathbf{f}_L\right) \propto pdf\left(\boldsymbol{\alpha}_L\right) \cdot pdf\left(\mathbf{f}_L \middle| \boldsymbol{\alpha}_L\right). \tag{16}$$

The prior distribution $pdf(\boldsymbol{\alpha}_L)$ is already defined in (13). To derive the likelihood function $pdf(\mathbf{f}_L \mid \boldsymbol{\alpha}_L)$, we further assume that the error for the late-stage performance model $f_L(\mathbf{x})$ follows a zero-mean Gaussian distribution and, hence, the approximate equality in (7) can be re-written as:

$$f_L(\mathbf{x}) = \sum_{m=1}^{M} \alpha_{L,m} \cdot g_m(\mathbf{x}) + \varepsilon_L \tag{17}$$

where ε_L denotes the modeling error with the distribution:

$$pdf(\varepsilon_L) = \frac{1}{\sqrt{2\pi} \cdot \sigma_0} \cdot \exp\left(-\frac{\varepsilon_L^2}{2 \cdot \sigma_0^2} \right) \sim N\left(0, \sigma_0^2\right). \tag{18}$$

In (18), the standard deviation σ_0 controls the magnitude of the modeling error. Its value can be optimally determined by using the cross-validation technique that will be discussed in Section 4.2.

Given (17)-(18), since the modeling error at the kth simulation sample $(\mathbf{x}^{(k)}, f_L^{(k)})$ is simply one sampling point of the random variable ε_L, it follows the Gaussian distribution:

$$f_L^{(k)} - \sum_{m=1}^{M} \alpha_{L,m} \cdot g_m\left(\mathbf{x}^{(k)}\right) \sim N\left(0, \sigma_0^2\right). \tag{19}$$

Therefore, the probability of observing the kth sampling point is:

$$pdf\left(f_L^{(k)} \middle| \boldsymbol{\alpha}_L\right) = \frac{\exp\left\{ -\frac{1}{2 \cdot \sigma_0^2} \cdot \left[f_L^{(k)} - \sum_{m=1}^{M} \alpha_{L,m} \cdot g_m\left(\mathbf{x}^{(k)}\right) \right]^2 \right\}}{\sqrt{2\pi} \cdot \sigma_0}. \tag{20}$$

Assume that all sampling points are independently generated, we can write the likelihood function $pdf(\mathbf{f}_L \mid \boldsymbol{\alpha}_L)$ as:

$$pdf\left(\mathbf{f}_L \middle| \boldsymbol{\alpha}_L\right) = \prod_{k=1}^{K} pdf\left(f_L^{(k)} \middle| \boldsymbol{\alpha}_L\right). \tag{21}$$

Combining (13), (16) and (20)-(21), it is straightforward to prove that the posterior distribution $pdf(\boldsymbol{\alpha}_L \mid \mathbf{f}_L)$ is Gaussian and its covariance matrix $\boldsymbol{\Sigma}_L$ and mean vector $\boldsymbol{\mu}_L$ are [17], [19]:

$$\boldsymbol{\Sigma}_L = \left[\sigma_0^{-2} \cdot \mathbf{G}^T \cdot \mathbf{G} + diag\left(\sigma_1^{-2}, \sigma_2^{-2}, \cdots, \sigma_M^{-2}\right) \right]^{-1} \tag{22}$$

$$\boldsymbol{\mu}_L = \sigma_0^{-2} \cdot \boldsymbol{\Sigma}_L \cdot \mathbf{G}^T \cdot \mathbf{f}_L \tag{23}$$

where $\mathbf{G}$ and $\mathbf{f}_L$ are defined by (3) and (15) respectively, and diag(•) represents an operator to construct a diagonal matrix. Since the Gaussian PDF $pdf(\boldsymbol{\alpha}_L \mid \mathbf{f}_L)$ reaches its maximum at the mean value, the MAP solution $\boldsymbol{\alpha}_L$ is equal to the mean vector $\boldsymbol{\mu}_L$:

$$\boldsymbol{\alpha}_L = \sigma_0^{-2} \cdot \boldsymbol{\Sigma}_L \cdot \mathbf{G}^T \cdot \mathbf{f}_L. \tag{24}$$

In other words, Eq. (24) shows the optimal coefficients solved by our proposed BMF method for the late-stage performance model $f_L(\mathbf{x})$.

While the basic idea of prior knowledge definition and maximum-a-posteriori estimation is illustrated in this section, several implementation issues must be carefully considered in order to make BMF of practical utility. These implementation details will be further discussed in the next section.

4. IMPLEMENTATION ISSUES

To make the proposed BMF method of practical utility, two implementation issues, (i) missing prior knowledge and (ii) cross-validation, must be carefully considered. In this section, we will discuss these implementation issues in detail.

4.1 Missing Prior Knowledge

The BMF method derived in Section 3 assumes that the early-stage model $f_E(\mathbf{x})$ and the late-stage model $f_L(\mathbf{x})$ share the same basis functions. In practice, this assumption may not always hold, because the early-stage model does not necessarily capture all the detailed behaviors of a circuit. For instance, it is well-known that layout parasitics will be added to the post-layout netlist (late stage) during layout extraction. The variations of these parasitics must be modeled by a number of new random variables that are completely ignored at the schematic level (early stage). The late-stage post-layout model $f_L(\mathbf{x})$ should contain additional basis functions corresponding to the new random variables that are not found from the early-stage schematic model $f_E(\mathbf{x})$. In this case, the early-stage model $f_E(\mathbf{x})$ does not carry any prior knowledge about the late-stage model coefficients associated with these additional basis functions. In other words, the prior knowledge for these late-stage model coefficients is missing.

To appropriately handle the cases with missing prior knowledge, we re-visit the prior distribution $pdf(\alpha_{L,m})$ defined in (8). As mentioned in Section 3.1, the standard deviation σ_m of the Gaussian distribution $pdf(\alpha_{L,m})$ encodes the magnitude information of the late-stage model coefficient $\alpha_{L,m}$. If there is no prior knowledge available for $\alpha_{L,m}$, it implies that the late-stage model coefficient $\alpha_{L,m}$ can possibly take any value with equal probability. Hence, the standard deviation σ_m should be set to $+\infty$:

$$\sigma_m = +\infty \tag{25}$$

so that the prior distribution is nearly constant over a wide range. Note that when calculating the posterior distribution in (22)-(23), only the value of σ_m^{-1} is needed. Hence, the infinite standard deviation in (25) would not cause any numerical problem for solving the late-stage model coefficients.

4.2 Cross-Validation

As mentioned in Section 3.2, the standard deviation σ_0 of the modeling error in (18) must be determined. Otherwise, without knowing σ_0, the late-stage model coefficients $\boldsymbol{\alpha}_L$ cannot be determined by the MAP solution in (24). The objective here is to find the optimal value of σ_0 so that the modeling error is minimized. Towards this goal, we must accurately estimate the modeling error for different σ_0 values and then select the optimal σ_0 with minimal error.

To quantitatively estimate the modeling error for a given σ_0 value, we adopt the idea of N-fold cross validation from the statistics community [19]. Namely, we partition the entire data set

into N groups. Modeling error is estimated from N independent runs. In each run, one of the N groups is used to estimate the modeling error and all other groups are used to calculate the model coefficients. Note that the training data for coefficient estimation and the testing data for error estimation are not overlapped. Hence, over-fitting can be easily detected. In addition, different groups should be selected for error estimation in different runs. As such, each run results in an error value e_n (n = 1, 2, ..., N) that is measured from a unique group of data points. The final modeling error is computed as the average of $\{e_n; n = 1, 2, ..., N\}$, i.e., $e = (e_1 + e_2 + ... + e_N)/N$. More details on cross-validation can be found in [19].

4.3 Summary

Algorithm 1 summarizes the major steps of our proposed BMF method. It consists of two core components: (i) prior distribution definition, and (ii) MAP estimation. The efficacy of BMF will be further demonstrated by our numerical examples in the next section.

Algorithm 1: Bayesian Model Fusion (BMF)

1. Start from the early-stage performance model $f_E(\mathbf{x})$ in (6).
2. Define the prior distribution for the late-stage model coefficients $\{\alpha_{L,m}; m = 1, 2, ..., M\}$ by (12)-(13) and (25).
3. Collect K late-stage simulation samples $\{(\mathbf{x}^{(k)}, f_L^{(k)}); k = 1, 2, \ldots, K\}$.
4. Solve the late-stage model coefficients $\{\alpha_{L,m}; m = 1, 2, ..., M\}$ based on (24) where σ_0 is determined by cross-validation.

5. NUMERICAL EXAMPLES

In this section, several circuit examples designed in a commercial 32nm CMOS process are used to demonstrate the efficacy of the proposed BMF method. Our objective is to build post-layout performance models for these circuits. For testing and comparison purposes, two different performance modeling techniques are implemented: (i) the traditional sparse regression method based on OMP [8], and (ii) the proposed BMF method. Here, the OMP algorithm is chosen for comparison, since it is one of the state-of-the-art techniques in the literature. When implementing BMF, we use the schematic-level simulation data to define our prior knowledge for post-layout performance modeling.

In each example, two different data sets, referred to as the training set and the testing set respectively, are generated by random sampling based on post-layout transistor-level simulation. The training set is used for coefficient fitting, including cross-validation. The testing set contains 300 independent random samples that are used for model validation. All numerical experiments are run on a 2.9GHz Linux server with 4GB memory.

5.1 Ring Oscillator

Shown in Figure 2(a) is the simplified circuit schematic of a ring oscillator designed in a commercial 32nm CMOS process. In this example, there are 7177 independent random variables in total to model device-level process variations, including both inter-die variations and random mismatches. Our objective is to approximate three post-layout performance metrics, power, frequency and phase noise, as linear functions of these 7177 random variables.

Figure 2(b)-(d) show how the performance modeling error varies with the number of post-layout training samples. Note that for both OMP and BMF, the modeling error decays as the number of samples increases. However, given the same number of post-layout training samples, BMF is able to achieve substantially higher accuracy than OMP, especially if only few samples are available.

Table 1 further compares the modeling error and cost for OMP and BMF. The total cost for performance modeling consists of two major portions: (i) simulation cost (i.e., the cost of running a transistor-level simulator to generate all post-layout samples in the training set), and (ii) fitting cost (i.e., the cost of solving all unknown model coefficients). As shown in Table 1, the total modeling cost is dominated by transistor-level simulation in this example. BMF achieves 9× runtime speed-up over OMP without surrendering any accuracy.

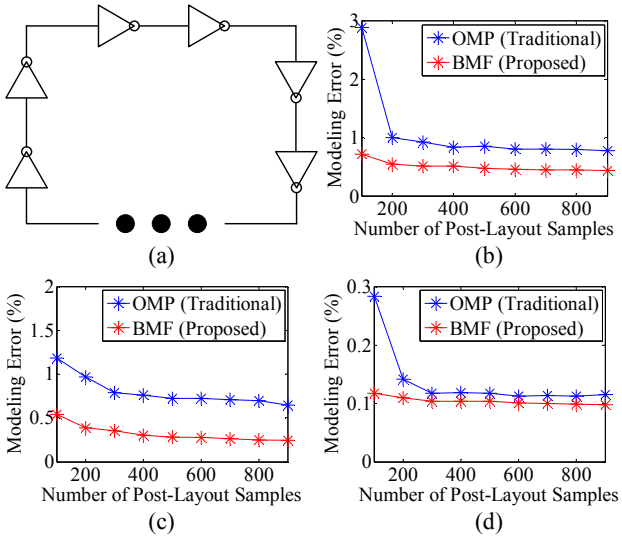

Figure 2. A ring oscillator designed in a commercial 32nm CMOS process is used as an example for performance modeling where BMF requires significantly less post-layout samples than OMP to achieve the same accuracy: (a) simplified circuit schematic of the ring oscillator, (b) modeling error for power, (c) modeling error for frequency, and (d) modeling error for phase noise.

Table 1. Performance modeling error and cost for ring oscillator

	OMP (Traditional)	BMF (Proposed)
# of post-layout training samples	900	100
Modeling error for power	0.77%	0.72%
Modeling error for frequency	0.65%	0.54%
Modeling error for phase noise	0.12%	0.12%
Simulation cost (Hour)	12.58	1.40
Fitting cost (Second)	5.75	1.69
Total modeling cost (Hour)	12.58	1.40

5.2 SRAM Read Path

Figure 3(a) shows the simplified circuit schematic of an SRAM read path designed in a commercial 32nm CMOS process. In this example, there are 66117 independent random variables to model device-level process variations. Read delay is our circuit performance of interest, and it is approximated as a linear function of all device-level random variables.

Figure 3(b) and Table 2 compare the modeling error and cost for OMP and BMF. Similar to the ring oscillator example, two important observations can be made here. First, BMF requires substantially less post-layout training samples than OMP, in order

to achieve the same modeling accuracy. In this example, BMF reduces the number of required samples from 400 to 100 without surrendering any accuracy. Figure 4 further plots the histograms of modeling error for both OMP and BMF with 400 and 100 post-layout training samples, respectively. Second, but more importantly, since the total modeling cost is dominated by transistor-level simulation, BMF successfully reduces the total modeling cost by reducing the number of required post-layout training samples. Compared to OMP, BMF achieves 4× runtime speed-up, as shown in Table 2.

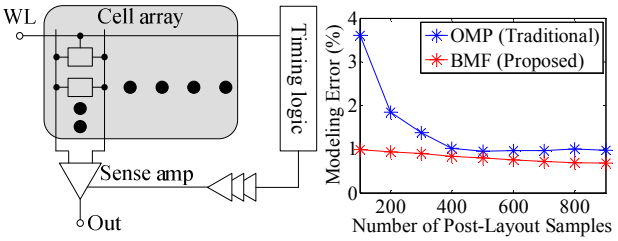

Figure 3. A simplified SRAM read path designed in a commercial 32nm CMOS process is used as an example for performance modeling where BMF requires significantly less post-layout samples than OMP to achieve the same accuracy: (a) simplified circuit schematic of the SRAM read path, and (b) modeling error for read delay.

Table 2. Performance modeling error and cost for SRAM

	OMP (Traditional)	BMF (Proposed)
# of post-layout training samples	400	100
Modeling error for read delay	1.02%	0.99%
Simulation cost (Hour)	38.77	9.69
Fitting cost (Second)	3.56	2.11
Total modeling cost (Hour)	38.77	9.69

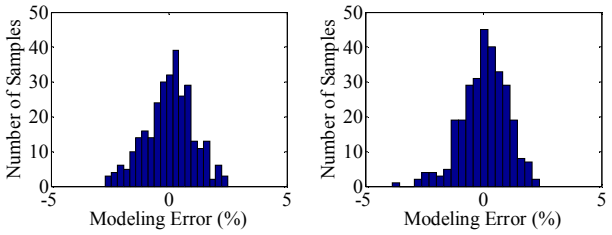

Figure 4. Histograms of modeling error are estimated from the testing set for read delay: (a) modeling error of OMP with 400 post-layout training samples, and (b) modeling error of BMF with 100 post-layout training samples.

6. CONCLUSIONS

In this paper, a novel BMF algorithm is proposed for efficient high-dimensional performance modeling of complex AMS circuits with consideration of large-scale process variations. BMF borrows the early-stage (e.g., schematic-level) simulation data to learn a model template that is statistically represented as a prior distribution. Next, the model template encoding our prior knowledge is further calibrated by very few late-stage (e.g., post-layout) simulation samples to accurately create a late-stage performance model. As such, the computational cost of high-dimensional performance modeling can be substantially reduced, since only few transistor-level simulations are required at the late stage. As is demonstrated by our circuit examples designed in a commercial 32nm CMOS process, the proposed BMF method achieves up to 9× runtime speedup compared to the traditional modeling technique. In our future work, we will further apply BMF to several practical applications such as statistical analysis of large-scale AMS systems.

7. ACKNOWLEDGEMENTS

This work has been supported in part by National Science Foundation and Intel Corporation.

8. REFERENCES

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

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

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

[7] T. McConaghy, "High-dimensional statistical modeling and analysis of custom integrated circuits," *IEEE CICC*, 2011.

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

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

[10] 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. on CAD*, vol. 24, no. 9, pp. 1372-1380, 2005.

[11] 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.

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

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

[14] 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.

[15] 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.

[16] X. Li, W. Zhang, F. Wang, S. Sun and C. Gu, "Efficient parametric yield estimation of analog/mixed-signal circuits via Bayesian model fusion," *IEEE ICCAD*, 2012.

[17] S. Ji, Y. Xue, and L. Carin, "Bayesian compressive sensing," *IEEE Trans. on Signal Processing*, pp. 2346-2356, Jun. 2008.

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

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