Sparse Statistical Model Inference for Analog Circuits under Process Variations

Yan Zhang, Sriram Sankaranarayanan and Fabio Somenzi
ECEE, University of Colorado, Boulder, CO, USA
{yan.zhang, srirams, fabio}@colorado.edu

Abstract— In this paper, we address the problem of performance modeling for transistor-level circuits under process variations. A sparse regression technique is introduced to characterize the relationship between the process parameters and the output responses. This approach relies on repeated simulations to find polynomial approximations of response surfaces. It employs a heuristic to construct sparse polynomial expansions and a stepwise regression algorithm based on LASSO to find low degree polynomial approximations. The proposed technique is able to handle many tens of process parameters with a small number of simulations when compared to an earlier approach using ordinary least squares. We present our approach in the context of statistical model inference (SMI), a recently proposed statistical verification framework for transistor-level circuits. Our experimental evaluation compares percentage yields predicted by our approach with Monte-Carlo simulations and SMI using ordinary least squares on benchmarks with up to 30 process parameters. The sparse-SMI approach is shown to require significantly fewer simulations, achieving orders of magnitude improvement in the run times with small differences in the resulting yield estimates.

I. Introduction

Nano-CMOS technologies are widely used in modern VLSI design. While the move to nano-regime brings benefits to designers, many challenges emerge [1]. Among these challenges, one of the most critical issues is to design robust circuits under increasingly large *process variations*.

Statistical performance modeling is a common approach to understanding the effects of process variations. It has been studied extensively in the past to characterize the performance of circuits in terms of process parameters [2–6]. However, these techniques can only address problems in older technologies, wherein process variations were not a dominant issue and could be modeled by a few “inter-die” parameters using linear models. In nanoscale era, the process parameters have to be treated in a *per-transistor* fashion. For instance, the gate oxide thickness has to be considered separately for each transistor [1]. Furthermore, the variations are becoming larger and require nonlinear modeling techniques [7]. These challenges call for an approach that is able to efficiently construct nonlinear approximations in the face of many tens or hundreds of process parameters.

Due to the design challenges in nano-regime, performance modeling has received renewed attention. Li *et al.* [8] proposed a PROBE approach based on reduced rank regression. Their technique enables the use of quadratic models for nonlinear response surfaces modeling. A similar idea was proposed by Feng *et al.* [9] to handle the problem of interconnect modeling. One drawback of these two techniques is that they are limited to quadratic models, and thus cannot characterize higher-order response surfaces accurately. Singhee *et al.* [10] proposed a nonlinear regression approach based on latent variable regression and neural networks. For many practical circuits, they showed promising results. However, due to the complicated structure of neural networks, the resulting models can be prone to over-fitting. Furthermore, neural network models are hard to interpret. Li [11] used matching pursuit to find the “best” projection of the response surface onto an orthogonal polynomial basis. Although in principle, their technique can handle higher-order response surfaces, the experimental results are limited to quadratic models.

In this paper, we present a sparse regression technique to discover a polynomial approximation of an output response as a function of input process parameters. Such an approximation is used inside a statistical framework to construct a mathematical model that explains how the process variations affect the output response. Our approach collects data by sampling process parameters and simulating the circuit to find the output responses. Using the simulation data, we construct a low-degree and sparse polynomial in a stepwise fashion. The proposed technique consists of a heuristic that discovers relevant terms in the polynomial approximation, and a sparse regression algorithm based on LASSO to construct the actual approximation [12]. The heuristic efficiently discards terms that contribute the least to the output response. Then the regression algorithm computes the coefficients for the remaining terms under $\mathcal{L}_1$ regularization. This technique is used inside *statistical model inference* (*SMI*),

This work was partially supported by the US National Science Foundation (NSF) under the award number 1016994 and CCF-1320069. All opinions expressed are those of the authors, and not necessarily of the NSF.

978-1-4799-2816-3/14/$31.00 ©2014 IEEE

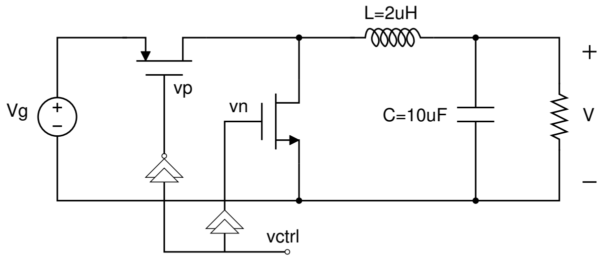

Fig. 1. A buck converter with L and C as process parameters.

a recently proposed statistical verification framework for transistor-level circuits [13]. SMI addresses the problem of estimating a safe region of the parameter space that guarantees a given design specification of a circuit, and examines specifications of the form $\phi(\mathbf{p}) \in [\ell, u]$ where ϕ is a performance metric, $\mathbf{p}$ a vector of process parameters, and ℓ and u are real-valued performance limit. We show that the proposed algorithm improves the scalability of SMI to handle circuits with many process parameters.

The rest of this paper is organized as follows. First, we provide an overview of SMI with a running example. Then we introduce the sparse regression algorithm in Section III. Finally, we demonstrate our technique by showing the verification of a ring oscillator and an operational amplifier under a large number of process variations.

II. Overview of SMI

In this section, we review the statistical verification framework SMI with a running example [13]. Consider a circuit with process parameters $\mathbf{p}$ from a parameter space $\mathbb{P}$ with a joint distribution $\mathcal{F}$. Let $\phi = f(\mathbf{p})$ be a *output response* of interest. SMI handles design specifications of the form: $\phi \in [\ell, u]$, where ℓ and u are the tolerance limits of ϕ, and shows in the statistical sense whether a specification is satisfied under process variations.

Example II.1 (Buck Converter)**.** *Figure 1 shows a simple buck converter. Assume that L and C are independent, uniform random variables:* [1] *$L \in [1.8, 2.2]\mu H$ and $C \in [9, 11]\mu F$. We verify that $\Delta v \leq 30mV$, where Δv is the ripple amplitude of the output voltage. For simplicity, the transistors are treated as ideal switches. Then we have*

$$\Delta v = f(L, C) = \frac{V_g - V}{16LC} DT_s^2 \,, \qquad (1)$$

where V is the DC component of the output voltage, D is the duty cycle and T_s is the time period of the control voltage. Let $V = 3\,V$, $D = 0.25$ and $T_s = 2\mu s$. From (1), we derive the condition $LC \geq 18.75$ that guarantees the satisfaction of the specification.

In practice, ϕ is usually represented by a complicated function f with a unknown closed from. To verify a given specification, SMI relies on repeated simulations to find a *statistical over-approximation* of f, which consists of a polynomial q and a tolerance interval I such that

$$\mathsf{Pr}\left(\phi \in q(\mathbf{p}) \oplus I\right) \geq \theta \,,\ \mathbf{p} \in \mathbb{P} \,, \qquad (2)$$

where $\oplus$ denotes the Minkowski sum and θ is a given probability. This means that the pair (q, I) over-approximates ϕ for at least θ portion of the parameter space $\mathbb{P}$. Such an approximation can be used to derive a *statistically* safe subset of $\mathbb{P}$ for the specification.

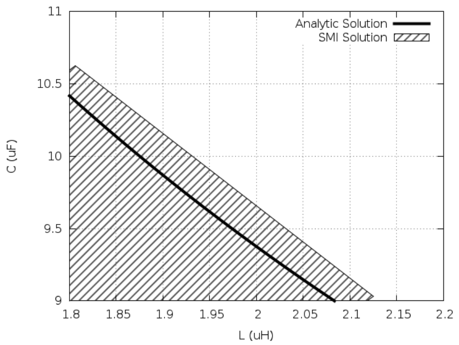

Fig. 2. Verification result of the buck converter. The region below the solid curve is shown to be unsafe. The shaded region is the statistically unsafe subset predicted by SMI.

To obtain a statistical over-approximation of ϕ, SMI works in two phases: 1) a regression phase that finds a polynomial q to approximate ϕ, and 2) a bloating phase that derives a tolerance interval I. In the first phase, we construct a polynomial approximation q. A simple approach based on ordinary least squares (OLS) is used in Zhang *et al.* [13], which does not scale well to circuits with many parameters. Therefore, a novel algorithm is presented in Section III to improve SMI by using sparse regression techniques. The bloating phase remains unaltered and is briefly presented below.

Bloating. With an approximation q, we consider the problem of deriving a tolerance interval I for a given probability θ such that (2) holds. In SMI, it is formulated as a hypothesis testing problem that involves two hypotheses:

$$\mathcal{H}_0 : \mathsf{Pr}(\phi \in q \oplus I) \geq \theta \text{ and } \mathcal{H}_1 : \mathsf{Pr}(\phi \in q \oplus I) < \theta \,.$$

We wish to find an interval I so that we can be convinced of $\mathcal{H}_0$ as opposed to $\mathcal{H}_1$. In SMI, the interval I is derived using an algorithm based on Bayesian hypothesis testing. Starting with a zero interval $I = [0, 0]$, SMI repeatedly simulates the circuit with randomly sampled parameters and adjusts the interval I until for some $K > 0$ consecutive simulations, $\phi \in q \oplus I$. The threshold K is chosen so that when it is achieved, the observed data provide strong statistical evidence to accept $\mathcal{H}_0$, indicating that the interval I is sufficiently large. The rationale of SMI including the choice of K is discussed by Zhang *et al.* [13].

The two steps, regression and bloating, construct a model (q, I). Intuitively, such a model convinces a statistical model checker [14, 15] that (2) holds with a given

[1] Our approach can handle a larger variety of distributions that one may encounter in practice.

θ. With this model, identifying the safety of a point $\mathbf{p}$ reduces to checking whether $q(\mathbf{p}) \oplus I \subseteq [\ell, u]$. Furthermore, S, the safe subset of the parameter space, can be computed conservatively by nonlinear solvers, such as Z3 [16] and iSAT [17], or statistically by Monte-Carlo sampling.

Example II.2 (Verifying Buck Converter). *We use $N = 20$ simulations to construct a cubic polynomial $q_3(L, C)$. Using OLS, we obtain the following polynomial* [2].

$$\begin{pmatrix} 0.031 - 0.002L - 0.009C - 0.002L^2 - 0.005C^2 + 0.003LC + \\ 0.004L^2C - 0.002LC^2 - 0.000L^3 + 0.010C^3 \end{pmatrix} . \quad (3)$$

As a comparison, using the proposed algorithm (as described below), we obtain a sparser approximation

$$0.028 - 0.001L - 0.001C \, . \quad (4)$$

Clearly, (4) *provides a more compact representation than* (3). *Such a polynomial is bloated with $\theta = 0.95$, leading to a tolerance interval $I = [-3, 4]mV$. The model (q_3, I) is used to check the specification, $\Delta v \leq 30mV$. The result is shown in Figure 2. The solid curve plots of the equation $LC = 18.75$ so that the region below is known to be unsafe. The shaded region is the unsafe subset predicted by SMI. It can be shown that in this example, bloating* (4) *yields identical verification result as bloating* (3).

III. A Sparse Regression Algorithm

In this section, we introduce a sparse regression algorithm. Let $\mathbf{p}$ be the process parameters and ϕ be a response of a circuit. Let $w(\mathbf{p})$ be the probability density function of $\mathbf{p}$. Our goal is to find a polynomial to approximate the response surface using a limited number of samples. The main challenge is that the number of parameters is often large, while the number of simulations is greatly limited by the available computational power. This means that the underlying linear system is under-determined. Using conventional approaches, such as least squares fitting, can easily lead to over-fitting.

To solve this problem, we propose an approach shown in Algorithm 1, which uses LASSO [12], a $\mathcal{L}_1$ regularization technique, to solve the under-determined system. Let $\phi = f(\mathbf{p})$ where f is a square-integrable function with an unknown closed form. Given a set of n samples, $\mathbf{X}$ is an $n \times k$ matrix where k is the number of parameters, and $\mathbf{Y}$ consists of n observations of the property ϕ. The algorithm operates in an iterative fashion, wherein the approximation is constructed using a sequence of *latent variables.* The j-th iteration attempts to fit a polynomial of degree j to approximate the *residue* $\mathbf{R}_{j-1}$ from the previous iteration in terms of $\mathbf{X}$. Initially $\mathbf{R}_0 = \mathbf{Y}$. At each step, we discover a latent variable $z_j = q_j(\mathbf{p})$ that provides an approximation q_j of $\mathbf{R}_{j-1}$, and $\mathbf{R}_j = \mathbf{R}_{j-1} - q_j(\mathbf{p})$. The process is stopped when the degree limit D is reached or when $|R_j|$ is below a threshold. As a final step, the overall model is constructed as a polynomial that approximates $\mathbf{Y}$ in terms of the latent variables $z_1, \ldots, z_k$. Our approach finds a hierarchical model that expresses the output response ϕ as a function of latent variables $z_1, \ldots, z_k$, each of which is a polynomial of increasing degree over $\mathbf{p}$.

```
Data: Matrices $\mathbf{X}, \mathbf{Y}$, Degree Limit $D$, Initial Degree $d$
Result: Approximation $q(\mathbf{p})$
latent_vars := empty set ;
$\mathbf{R} := \mathbf{Y}$ ;
while true do
    $S'_d$ := choose_a_sparse_basis($d, \mathbf{X}, \mathbf{R}$) ;
    $z, \mathbf{R}$ := compute_latent_variables_by_lasso($S'_d, \mathbf{X}, \mathbf{R}$) ;
    latent_vars := latent_vars $\cup\ z$ ;
    $d := d + 1$ ;
    if $|\mathbf{R}| < \epsilon$ or $d == D$ then
        $q(\mathbf{p})$ := do_least_squares_fitting(latent_vars, $\mathbf{X}, \mathbf{Y}$) ;
    end
end
```

Algorithm 1: Sparse Regression Algorithm.

A. Choosing a Sparse Basis

Polynomial chaos expansion (PCE) provides a mathematical basis for approximating functions of random variables by a polynomial obtained by a truncated generalized Fourier series

$$f(\mathbf{p}) \approx \sum_{i \in S_d} c_i Q_i(\mathbf{p}) \, ,$$

where S_d is an index set of terms, c_i a generalized Fourier coefficient, and $\{Q_i \mid i \in S_d\}$ is a set of orthogonal polynomials of degree d [18]. The polynomials Q_i are chosen based on the distribution $\mathcal{F}$ of process parameters using the standard Wiener-Askey scheme. We call each Q_i as a *term.* Intuitively, if c_i is sufficiently small, the corresponding term contributes little to f and can be dropped without losing much accuracy. The coefficients c_i can be computed from the following equation [18].

$$c_i = \frac{1}{\gamma_i} \langle f, Q_i \rangle = \frac{1}{\gamma_i} \int_{\mathbf{p} \in \mathbb{P}} f(\mathbf{p}) Q_i(\mathbf{p}) w(\mathbf{p}) d\mathbf{p}$$

where each γ_i is a normalization constant, $\langle , \rangle$ an inner product, and $w(\mathbf{p})$ a probability density function.

It can be difficult, if not impossible, to compute c_i exactly because (1) f may not have a closed form and (2) $\mathbf{p}$ may consist of a large number of parameters. As a possible solution, numerical quadrature methods suffer from their scalibility. They also rely on the ability to sample at certain points, which is not always realizable in practice.

We use Monte-Carlo methods to estimate c_i,

$$\hat{c}_i = \frac{1}{\gamma_i n} \sum_{j=1}^{n} f(\mathbf{p}_j) Q_i(\mathbf{p}_j) \, , \quad (5)$$

where n is the number of samples and each $\mathbf{p}_j$ is sampled according to $w(\mathbf{p})$. Note that such an estimation can at best capture the trend of c_i due to finite sample

[2] The parameter values for L and C are normalized to $[-1, 1]$.

size. Using $\hat{c}_i$ usually leads to poor approximations of the response surface. However, the computation of (5) is efficient even with many parameters. Thus in our approach, we use (5) to construct a *filtering stage* such that terms with small $\hat{c}_i$ are dropped. In particular, we drop those terms whose corresponding $\hat{c}_i$ are in the lower k-quantile of the whole set, where k is controlled by the user. The index set of remaining terms is denoted by S'_d.

B. Computing Latent Variables by LASSO

We need to find an approximation consisting of Q_i, $i \in S'_d$. This is usually solved as a least squares problem

$$\min_{c_i, i \in S'_d} \|f(\mathbf{p}) - g(\mathbf{p})\|_2^2 \ , \ g(\mathbf{p}) = \sum_{i \in S'_d} c_i Q_i(\mathbf{p}) \, . \qquad (6)$$

To avoid over-fitting, we use LASSO to solve the minimization problem in (6). LASSO handles the problem by adding an extra constraint on the coefficients c_i,

$$\min_{c_i, i \in S'_d} \|f(\mathbf{p}) - g(\mathbf{p})\|_2^2 + \alpha \sum_{i \in S'_d} |c_i| \ , \qquad (7)$$

where α is a Lagrangian multiplier of the regularization term $\sum_{i \in S'_d} |c_i|$. Intuitively, the extra term forces the coefficients c_i to behave "regularly" so that they cannot range over many orders of magnitude. Furthermore, due to the nature of $\mathcal{L}_1$-norm minimization, proper choices of α result in sparse solutions. In general, a larger α leads to a sparser solution. When α approachs 0, LASSO reduces to ordinary least squares regression.

To apply LASSO in our algorithm, we construct a $n \times |S'_d|$ matrix $\mathbf{X}_{exp}$ that comprises observations mapped to each expansion term,

$$\mathbf{X}_{exp} = \begin{pmatrix} Q_{i_1}(\mathbf{X}_1) & \cdots & Q_{i_m}(\mathbf{X}_1) \\ Q_{i_1}(\mathbf{X}_2) & \cdots & Q_{i_m}(\mathbf{X}_2) \\ \vdots & \ddots & \vdots \\ Q_{i_1}(\mathbf{X}_n) & \cdots & Q_{i_m}(\mathbf{X}_n) \end{pmatrix} \ , \ i_k \in S'_d \, , \qquad (8)$$

where $\mathbf{X}_i$ is the i-th row of $\mathbf{X}$. Denote the coefficient vector $\mathbf{c} = (c_{i_1}, \ldots, c_{i_m})^T$. We solve the LASSO problem

$$\min_{\mathbf{c}} \|\mathbf{R}_{j-1} - \mathbf{X}_{exp}\mathbf{c}\|_2^2 + \alpha |\mathbf{c}| \, ,$$

where $\mathbf{R}_{j-1}$ is the residue from the previous iteration. The solution $\mathbf{c}$ yields a latent variable

$$z = c_{i_1} Q_{i_1} + \cdots + c_{i_m} Q_{i_m} \ , \ i_j \in S'_d \, ,$$

and the updated residue $\mathbf{R}_j = \mathbf{R}_{j-1} - \mathbf{X}_{exp}\mathbf{c}$.

C. Least Squares Fitting with Latent Variables

The iteration in Algorithm 1 terminates when either the residue $\mathbf{R}$ is smaller than some given ϵ, or the degree d reaches the limit D. The resulting model consists of the computed latent variables, each represented by a polynomial. In practice, since the number of latent variables is usually small, the fitting can be easily performed.

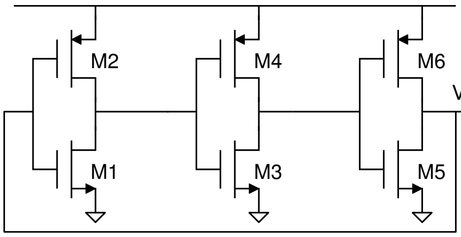

Fig. 3. A three-stage ring oscillator.

IV. Experimental Evaluation

In this section, we experimentally evaluate the sparse-SMI approach proposed in this paper, comparing it with the original SMI proposed by Zhang *et al.* [13] and a yield estimation using Monte-Carlo simulation. Our comparison involves two analog circuits: a three-stage ring oscillator and a two-stage operational amplifier (opamp) over a range of output response functions (properties) for these circuits. The sparse-SMI technique presented in this paper is implemented in Python. We use LTSpice [19] as the circuit simulator. All experiments were performed on a quad-core 2.8GHz machine running Debian 6.0. Unless specified, all the times are measured in seconds.

The comparisons between SMI and sparse-SMI are done through a $\theta = 0.95$ and a threshold $K = 89$, indicating that the tolerance intervals covers at least 95% of the parameter space with a high statistical confidence. We compare the resulting models by using them to predict the yield, which is also computed using Monte-Carlo Simulations for comparison.

A. Ring Oscillator

Figure 3 shows the schematic of a three-stage ring oscillator. The circuit is designed to oscillate at 0.98GHz with a tolerance within ±50MHz. The oscillation frequency is affected by various process parameters in the transistors $M_i \, , i = 1, \ldots, 6$. We select 24 process parameters to study, including the oxide thickness t_{ox}, threshold voltage under zero bias v_{t0}, channel width w and channel length l for each M_i. We assume that each variation follow a *truncated normal distribution* in the range $[\mu - 3\sigma, \mu + 3\sigma]$ where μ is the nominal value. The standard deviation of each variation is summarized in Table I.

TABLE I
Standard deviation of each process parameter. μ is the nominal value of the corresponding parameter.

	$t_{ox\,i}$	$v_{t0\,i}$	w_i	l_i
σ	0.05μ	0.05μ	0.1μ	0.1μ

Experiment results. The experiment results for the oscillator are presented in Table II. We choose a degree limit of 3 for SMI and sparse-SMI. The results for 10000 Monte-Carlo simulations are included. The

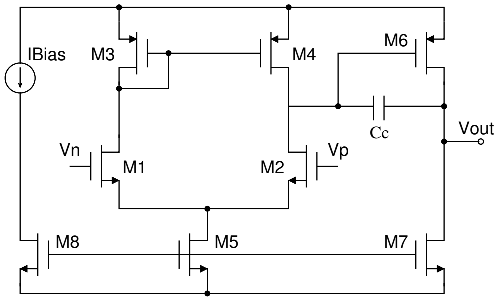

Fig. 4. A two-stage operational amplifier.

columns "Yield", "Var" and "SimTime" under "Monte-Carlo" show the sampling yield, variance of the sampling yield and time spent on simulations. Under the columns "SMI" and "Sparse-SMI", "#Sims" refers to the number of simulations performed in the regression and the bloating phase, respectively. Similarly, "SimTime" shows the simulation time and "Time" indicates the time spent in the regression and bloating phases of SMI, excluding the simulation time. "Safe" reports whether the specification is satisfied and if not, the predicted yield under the given variations. "Degree" is the final degree of the approximation in sparse-SMI.

Table II shows that under the given process variations, the specification $f_{osc} \in [0.93, 1.03]$MHz is not satisfied. First, let us compare SMI and sparse-SMI with the Monte-Carlo method. Since the SMI procedure procedure builds a statistically sound over-approximation for the underlying response surface, it tends to under-estimate the yield. It explains why the predicted yields from SMI and sparse-SMI are lower than the sampling yield from Monte-Carlo simulations. The results clearly show the advantage of using sparse-SMI as opposed to SMI. Whereas SMI requires 3200 simulations, sparse-SMI requires roughly 400 simulations in this case to result in almost identical yield estimates. The results reduce simulation time from roughly 1.5 hours for SMI to 12 minutes for sparse-SMI, a 7.5× reduction. Comparing to Monte-Carlo simulations also show the benefits of a model building approach rather than direct yield estimation.

B. Two-stage Operational Amplifier

The schematic of a commonly used two-stage opamp is shown in Figure 4. The performance of an opamp is characterized by many properties, such as input offset voltage and slew rate. Usually, each property is measured using a specific type of simulation and circuit configuration. For example, input offset voltage is often measured by arranging the opamp in the unity-gain configuration and sweeping DC input voltage. In contrast, the measurement of slew rate requires transient simulation. Table III shows a list of specifications under verification and the types of simulation for the measurement. For a detailed description on how to simulate these properties, we refer the interested reader to [20] (Chapter 6.6).

TABLE III
A list of opamp specifications under verification.

ID	Specification	Sim type
1	Input offset voltage $\leq$ 50mV	DC
2	DC voltage gain $\geq$ 60dB	AC
3	Unity-gain bandwidth $\geq$ 5MHz	AC
4	Phase margin $\geq 30°$	AC
5	CMRR $\geq$ 80dB	AC
6	PSRR (+) $\geq$ 80dB	AC
7	Slew rate (+) $\geq$ 10V/μs	Transient

We select 4 process parameters, the oxide thickness t_{ox}, threshold voltage under zero bias v_{t0}, channel width w and channel length l, for each transistor $M_i, i = 1, \ldots, 8$. It leads to a total of 32 parameters. We assume the same variations for each parameter as in Table I.

Experiment results. The experimental results for the opamp are summarized in Table IV. The first column, "ID", indicates the specification under verification. Again, we include 10000 Monte-Carlo simulations for comparison. The degree limit for SMI and sparse-SMI is 3. For sparse-SMI, we use 200 simulations in the regression phase for all the properties.

First, we compare SMI and sparse-SMI with the Monte-Carlo method. As in the ring oscillator experiment, we observe that the predicted yield from the two approaches are similar. For all the properties, sparse-SMI requires fewer simulations and finishes much faster. Sparse-SMI reduces the running times from many hours to a few minutes, often resulting in 25× or larger speed-ups.

The speed-up in terms of running time and number of simulations required is much more significant than in the ring oscillator verification. The regression time in SMI has become prohibitively large, while that in sparse-SMI is quite affordable. This demonstrates the scalability of sparse-SMI to handle problems that are too large for conventional approaches.

In the "Degree" column, we see that for some properties, sparse-SMI is able to construct accurate models with a degree lower than the limit. Correspondingly, for these properties, the regression times are significantly smaller. In fact, the major cost in regression lies in the computation of the generalized Fourier coefficients and LASSO. The former can be easily parallelized, leading to further performance improvements.

V. Conclusion

In this paper, we propose a sparse regression algorithm to address the problem of performance modeling for analog circuits under process variations. The algorithm uses a limited number of simulations to approximate response surfaces with many parameters. We have applied this technique to statistical model inference, a recently proposed statistical verification framework that aims to construct statistical over-approximations of the response sur-

TABLE II
Verification results for the ring oscillator with 24 process parameters.

Monte-Carlo		SMI				Sparse-SMI				
Yield	SimTime	#Sims	SimTime	Time	Safe	#Sims	SimTime	Time	Degree	Safe
51%	4.8h	3000/213	1.5h	391s/17s	×, 46%	200/197	693s	15s/4s	3	×, 45%

TABLE IV
Verification results for the opamp with 32 process parameters.

ID	Monte-Carlo		SMI				Sparse-SMI				
	Yield	SimTime	#Sims	SimTime	Times	Safe	#Sims	SimTime	Time	Degree	Safe
1	61%	1.8h	8500/201	1.6h	4.2h/45s	×, 58%	200/193	264s	5s/3s	1	×, 58%
2	65%	1.4h	8500/118	1.3h	4.9h/29s	×, 61%	200/180	191s	96s/5s	3	×, 60%
3	100%	1.9h	8500/241	1.7h	5.1h/56s	✓	200/161	271s	4s/3s	1	✓
4	98%	1.9h	8500/197	1.7h	4.4h/47s	×, 94%	200/131	229s	101s/4s	3	×, 93%
5	100%	2.2h	8500/194	1.7h	3.9h/51s	✓	200/281	381s	34s/7s	2	✓
6	62%	2.1h	8500/156	1.8h	4.9h/33s	×, 55%	200/154	260s	91s/5s	3	×, 54%
7	87%	2.8h	8500/123	2.5h	4.1h/28s	×, 85%	200/201	413s	6s/5s	1	×, 84%

face. We show that our sparse regression algorithm can significantly improve the scalability of SMI to handle circuits with many process parameters.

References

[1] L. L. Lewyn, T. Ytterdal, C. Wulff, and K. Martin, "Analog circuit design in nanoscale CMOS technologies," *Proceedings of the IEEE*, vol. 97, no. 10, pp. 1687–1714, 2009.

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

[3] G. Debyser and G. Gielen, "Efficient analog circuit synthesis with simultaneous yield and robustness optimization," in *ICCAD 98*, 1998, pp. 308–311.

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

[5] H. Yoon, P. Variyam, A. Chatterjee, and N. Nagi, "Hierarchical statistical inference model for specification based testing of analog circuits," in *VLSI Test Symposium*, 1998, pp. 145–150.

[6] H. Chang and S. S. Sapatnekar, "Statistical timing analysis considering spatial correlations using a single PERT-like traversal," in *ICCAD*, 2003, p. 621.

[7] S. R. Nassif, "Modeling and analysis of manufacturing variations," in *IEEE CICC*, 2001, pp. 223–228.

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

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

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

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

[12] R. Tibshirani, "Regression shrinkage and selection via the LASSO," *Journal of the Royal Statistical Society*, pp. 267–288, 1996.

[13] Y. Zhang, S. Sankaranarayanan, F. Somenzi, X. Chen, and E. Ábraham, "From statistical model checking to statistical model inference: Characterizing the effect of process variations in analog circuits," in *ICCAD*, 2013, preprint available on-line at http://bit.ly/12TH1WR.

[14] H. L. S. Younes and R. G. Simmons, "Probabilistic verification of discrete event systems using acceptance sampling," in *CAV*, 2002, pp. 223–235.

[15] S. K. Jha, E. M. Clarke, C. J. Langmead, A. Legay, A. Platzer, and P. Zuliani, "A Bayesian approach to model checking biological systems," in *CMSB*, 2009, pp. 218–234.

[16] L. De Moura and N. Bjørner, "Z3: an efficient smt solver," in *TACAS*, 2008, pp. 337–340.

[17] M. Fränzle, C. Herde, T. Teige, S. Ratschan, and T. Schubert, "Efficient solving of large non-linear arithmetic constraint systems with complex boolean structure," *JSAT*, vol. 1, no. 3-4, pp. 209–236, 2007.

[18] D. Xiu, *Numerical Methods for Stochastic Computation: A Spectral Method Approach.* Princeton university Press, 2010.

[19] "LTSpice." [Online]. Available: http://www.linear.com/designtools/software/

[20] P. E. Allen and D. R. Holberg, "CMOS analog circut design." Oxford University Press, Inc., 2002.