# Exploiting Correlation Kernels for Efficient Handling of Intra-Die Spatial Correlation, with Application to Statistical Timing

Amith Singhee, Sonia Singhal, Rob A. Rutenbar
Carnegie Mellon University, Pittsburgh, PA, USA
{asinghee,soniasin,rutenbar}@ece.cmu.edu

**Abstract**

Intra-die manufacturing variations are unavoidable in nanoscale processes. These variations often exhibit strong spatial correlation. Standard grid-based models assume model parameters (grid-size, regularity) in an *ad hoc* manner and can have high measurement cost [1]. The random field model [1][2] overcomes these issues. However, no general algorithm has been proposed for the practical use of this model in statistical CAD tools. In this paper, we propose a robust and efficient numerical method, based on the Galerkin technique [3] and Karhunen Loéve Expansion [4], that enables effective use of the model. We test the effectiveness of the technique using a Monte Carlo-based Statistical Static Timing Analysis algorithm, and see errors less than 2.8%, while reducing the number of random variables from thousands to 25, resulting in speedups of up to 10x.

## 1 Introduction

The scaling of technologies toward the nanometer regime brings with it a challenging increase in the amount of variability we must model, manage, and optimize, across all phases of chip design. Variation sources may be global (e.g., wafer-level process problems) or local (e.g., random dopant variation in a single device), and possess a complex spatial or temporal correlation structure. These problems have generated a wave of new statistically "aware" tools and methods; e.g., Statistical Static Timing Analysis (SSTA) [2][5][6], variational power-grid analysis [7].

Manufacturing variations may be classified into two categories based on the source of variation: (1) *systematic* variation, and (2) *random* variation. Systematic variation constitutes the deterministic part of these variations; e.g., proximity lithography effects, nonlinear etching effects, etc. [9]. These are typically pattern dependent and can potentially be completely explained by using more accurate models. Random variations constitute the unexplained part of the manufacturing variations, and show stochastic behavior; e.g., oxide thickness variations and random dopant fluctuation [10]. In this paper, we focus on modeling and handling these random variations. The within-die component of these random variations often exhibit spatial correlation: devices close to each track each other better than devices far apart [11][12].

Recently developed SSTA tools have recognized these unavoidably random aspects of manufacturing variations and have attempted to account for them using simple models that are computationally inexpensive. [5][6] use a linear model for the gate delay as a function of varying gate parameters. [5][13][14] recognize the impact of spatial correlation between gates and use simple grid-based models to represent this correlation. [5] uses Principal Components Analysis (PCA) to extract uncorrelated parameters from this correlation model, and builds the gate timing models using these uncorrelated components.

However, the grid-based model is *ad hoc* due to the lack of theoretical rigor in constructing the model: are these models valid with respect to the actual physics of the variations? What should be the grid resolution? What should be the amount of correlation between different grid cells? [15] tries to answer the last question by proposing a Bayesian learning-based approach to estimating the correlation coefficients from measurements. Recently, [1] presented the criteria for a physically valid spatial correlation model and showed the practical problems with using a grid-based model. The paper proposed representing the random variations for each parameter (e.g. $V_t$, $t_{ox}$) as a two-dimensional *random field*, and provided a construction for a valid correlation model in the form of an isotropic *correlation function* (*correlation kernel*) $K(\mathbf{x},\mathbf{y})$ [1]. $K(\mathbf{x},\mathbf{y})$ returns the correlation between any two locations $(\mathbf{x}, \mathbf{y})$ on the chip. Similar kernels (*correlograms*) are extracted in [16]. However, they do not provide any method to effectively use this model in statistical CAD tools. [2] views the chip-wide variation as a stochastic process and proposes using the *Karhunen-Loéve Expansion* (KLE) to extract a small set of uncorrelated random variables, which can then be used as parameters for the gate timing models. However, the approach is restricted to a theoretically simple, but physically unrealistic, correlation kernel, and no generic method is proposed to handle any realistic kernel extracted from process data (e.g., as per [1]).

In this paper, we recognize that it is crucial to account for intra-die variation, including spatial correlation, in today's statistical tools and present a complete, general and robust numerical method to handle arbitrary (physically valid) spatial correlation kernels. We provide strong theoretical justification for the proposed method, along with the relevant convergence properties. The proposed technique uses a Galerkin method along with numerical integration to evaluate the KLE of any two-dimensional stochastic process. The rest of the paper will elaborate on the relevant background, details of the technique and corroborative experimental results.

The paper is organized as follows. Sec. 2 briefly reviews the relevant theory of random fields (stochastic processes over space) and KLE. Sec. 3 lays the theoretical basis for the proposed numerical method and Sec. 4 provides the details of the method itself. Sec. 5 presents our experimental setup and results, and Sec. 6 offers concluding remarks.

## 2 Spatial Correlation Models

### 2.1 Grid-based spatial correlation model

Consider a statistical device parameter $p_k$, for example the effective channel length $L$ of devices on a chip. Here we only consider the random component of manufacturing variations, assuming that the systematic component can be characterized and included in the mean of the parameter variation. Using the grid correlation model [5][13], the chip area $D$ is divided into a grid, and each grid cell $g_i$ is assigned its own random variable (RV) $p_{k,i}$ for the value of the parameter.

[1] We use bold letter to denote vector; e.g., $\mathbf{x} = \{x_1, y_1\}$

This approach results in $N_G$ RVs for each such device parameter, where $N_G$ is the number of grid cells, with an $N_G \times N_G$ correlation matrix $\mathbf{K}_k$ describing the correlation amongst the cells. To simplify explanation, let us assume that the device parameters are centered about their mean and scaled by their standard deviations ($p_k = (p_k - \mu_k)/\sigma_k$) such that the variance of each parameter is now 1. This normalization can be accounted for in the gate timing model. Then, the covariance and correlation matrices are identical, and we will refer to both by $\mathbf{K}_k$. As is common [5][2], we assume in this paper that parameters $p_k$ and $p_j$ vary independently for $k \neq j$. For clarity, we will now drop the subscript $k$ and recognize the implicit dependence. A typical approach is to perform an orthogonal decomposition on the covariance matrix $\mathbf{K}$, using Principal Components Analysis (PCA), to extract an orthogonal basis of uncorrelated RVs, such that

$$p = \sum_{j=1}^{r} \sqrt{\lambda_j} v_j p'_j, \quad r = N_G \tag{1}$$

Here, $\lambda_j$ is the $j$-th largest eigenvalue of $\mathbf{K}$ and $v_j$ is the corresponding eigenvector. $p'_j$ are the uncorrelated RVs. If the $p_k$ are jointly normal, $p'_j$ are statistically independent. Fewer eigenpairs can be used ($r \ll N_G$) to reduce the number of RVs while losing some accuracy. Furthermore, the uncorrelated RVs help simplify the computations in a typical SSTA algorithm [5], particularly for Gaussian distributions.

However, there are some serious drawbacks with this grid-based model. First, it is not clear if it will always result in a valid (positive semi-definite) correlation matrix. [1] tries to address this problem of always learning a valid correlation model, by proposing the use of a *grid-less covariance kernel-based* model. We will further elaborate on this model in the following subsection. Second, it is difficult to estimate the optimal (or even near-optimal) grid structure: the number of grid cells, and the homogeneity of the grid. The typical approach is to assume a regular grid with an empirically assumed resolution, so as the keep the chip measurement costs reasonable while still maintaining some accuracy. We now describe the covariance kernel based model, which overcomes these problems elegantly.

### 2.2 Grid-less random field model

Let $K(\mathbf{x},\mathbf{y})$ be a function that returns the covariance of the parameter $p$ (e.g. $L, V_t, t_{ox}$) at locations $\mathbf{x}$ and $\mathbf{y}$ on the chip area $D$ ($\mathbf{x},\mathbf{y} \in D$). Hence, we refer to this function as the *covariance kernel* for $p$. Having normalized the parameters, the covariance is equal to the correlation. We anticipate the correlation to drop off monotonically as we move away from any given point $\mathbf{x}$ on the chip. A valid covariance kernel on a domain $D \times D$ must be *non-negative definite* [4] (also called positive definite): for every finite subset $D_n \subset D$ (finite set of points on the chip), and every function $h(\mathbf{x})$ on $D_n$

$$\sum_{\mathbf{x},\mathbf{x}' \in D_n} K(\mathbf{x},\mathbf{x}')h(\mathbf{x})\overline{h}(\mathbf{x}') \geq 0 \tag{2}$$

where $\overline{h}$ indicates the complex conjugate. From this it follows that $K$ must be symmetric [4]: $K(\mathbf{x},\mathbf{y}) = K(\mathbf{y},\mathbf{x})$. Fig. 1(a) shows an example of such a covariance kernel over the normalized chip area $D = [-1,1] \times [-1,1]$. Her, we have fixed $\mathbf{x}$ to $\mathbf{0}$ and vary $\mathbf{y}$ over the entire chip: we can see that the correlation drops away as we move away from $\mathbf{x}$.

Armed with this knowledge of covariance kernels, we now review relevant concepts from stochastic processes, to enable a clear understanding of the Karhunen Loéve expansion (KLE). Consider all possible *outcomes* of the normalized channel length variation across a die. In any one particular outcome, we will obtain a full set values of $L$, one for each device on the die. These random values will follow the underlying covariance kernel $K$. Fig. 1(b) shows two possible outcomes. We can see that devices close to each other exhibit similar $L$ values because of correlation, but this correlation is negligible for devices that are far apart. We denote the entire space of all such possible *outcomes* by $\Omega$. Let $\theta$ be an element of $\Omega$: for example, $\theta$ can represent any one of the two outcomes shown in Fig. 1(b). Then, we can define the $L$ value as a function of both the particular stochastic outcome and the location on the chip. Generalizing, any statistical parameter $p$ is a function $p(\mathbf{x},\theta)$ defined over $D \times \Omega$. Mathematically, $p : D \times \Omega \to \mathbf{R}$, where $\mathbf{R}$ is the real line. Such a function $p(\mathbf{x},\theta)$ is a *stochastic process*. If $D$ is defined over $n$ spatial dimensions (as opposed to, say, temporal dimensions) the process is also called a *random field*. Hence, we can treat each intra-die statistical parameter ($L$, $V_t$, $t_{ox}$, $W$) along with its respective covariance kernel, as a stochastic process (random field). We have indulged in some simplification and abuse of terminology here to make the theory simple and accessible. Please refer to [4] for a more rigorous treatment.

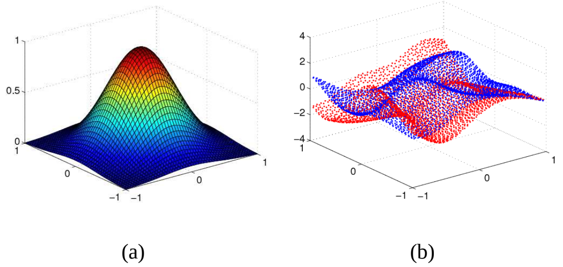


Figure 1: (a) A double exponential (Gaussian) covariance kernel. (b) Two possible outcomes of normalized $L$ values across the chip.

**Theorem 1** (Karhunen Loéve Expansion) *Let the stochastic process $p(\mathbf{x},\theta)$ on a closed domain $D$ have bounded variance over $D$ and a covariance kernel $K(\mathbf{x},\mathbf{y})$ that is continuous at $(\mathbf{x},\mathbf{x}), \forall \mathbf{x} \in D$. Then $p$ has the orthogonal decomposition*

$$p(\mathbf{x},\theta) = \sum_{j=0}^{\infty} \sqrt{\lambda_j} \xi_j(\theta) f_j(\mathbf{x}) \tag{3}$$

*where $\lambda_j$ is the $j$-th largest eigenvalue of the covariance kernel $K$ and $f_j(\mathbf{x})$ is the corresponding eigenfunction of $K$.*

The eigenpairs $(\lambda_j, f_j)$ are solutions of the integral equation

$$\int_D K(\mathbf{x},\mathbf{y}) f(\mathbf{y}) d\mathbf{y} = \lambda f(\mathbf{x}) \tag{4}$$

The eigenfunctions $f_j$ are orthonormal and the random variables $\xi_j$ are uncorrelated. For a proof, see [4]. From (3), we see that the $j$-th eigenvalue $\lambda_j$ is a measure of the amount of contribution made by the $j$-th RV $\xi_j$ to the overall variance of the process.

We note that all the statistical parameters of interest ($L$, $V_t$, etc.) do have bounded variance. Also, it is possible to have a physically valid covariance kernel that will be continuous at $(\mathbf{x},\mathbf{x})$: [1] outlines a robust method to extract such kernels. Hence, we can apply KLE to these statistical parameters. A truncation of the series in (3) yields an approximation to the process $p$, with a very useful property: it is optimal in the sense that it minimizes the mean squared error resulting from a finite representation of $p$. For a proof, see [8]. This implies that we can represent the infinitely large number of random variables spread over the domain $D$, using a finite, potentially small number of uncorrelated random variables $\xi_j$. In practice, the stochastic behavior of the thousands to millions of gates on a chip can potentially be compressed and represented using a much *reduced* set of *uncorrelated* random variables, enabling computationally efficient statistical CAD tools. Note the similarity between the PCA representation of (1) and the KLE representation of (3). This is not surprising given that PCA is a discrete form of KLE. In our problems of interest, the continuous correlation model and KLE are a more natural fit than enforcing an *ad hoc* discretization using a grid model and PCA.

## 3 Solution Method

### 3.1 Motivation

The usability of KLE hinges on the ability to solve the equation (4), and we propose a robust numerical technique for solving it. Equation (4) is a homogeneous Fredholm equation of the second kind, which has been studied extensively [3]. Analytical solutions to this equation are possible, but only for a few specific covariance kernels [8], and also often only for one-dimensional problems. These 1-D techniques can be extended to multi-dimensional problems if the kernel is separable into the product of analytically expandable 1-D kernels, as shown in [8]. The $j$-th eigenfunction (eigenvalue) of the separable kernel is then the product of the $j$-th eigenfunctions (eigenvalues) of the 1-D kernels. Even with this extension, analytical solutions are restricted to very few kernel forms. One example is the exponential kernel using the $L_1$ norm, written in two dimensions as

$$\begin{aligned} K(\mathbf{x},\mathbf{y}) &= e^{-c(|x_1-y_1|+|x_2-y_2|)} \\ &= k(x_1,y_1)k(x_2,y_2) = e^{-c|x_1-y_1|}e^{-c|x_2-y_2|} \quad (5) \end{aligned}$$

We see that it is separable into the product of two 1-D (identical) kernels. Analytical solutions for the latter are available [8]. However, this kernel is not practical as it uses the $L_1$ norm and the correlation decay behavior is unrealistic. [2] proposes using the kernel $\exp(-c|r_x - r_y|)$, where $r_x$, $r_y$ are the magnitudes ($L_2$ norm) of the vectors $\mathbf{x}$ and $\mathbf{y}$, so as to directly use the analytical solution for the 1-D exponential kernel. This kernel, too, is unrealistic as all points lying on an origin-centric circle will be perfectly correlated, even though the distance between them is large. Given measurement data, [1] proposes a technique to extract valid kernels of the form

$$K(\mathbf{x},\mathbf{y}) = 2\left(\frac{bv}{2}\right)^{s-1} B_{s-1}(bv)\Gamma(s-1)^{-1}, v = ||\mathbf{x}-\mathbf{y}||_2 \quad (6)$$

where we use $B$ to denote the modified Bessel function of the second kind, $\Gamma$ is the gamma function, and $b$ and $s$ are two real-valued shape parameters. Analytical solutions for these kernels are not known. Hence, we see the need for a generic numerical solution technique. We provide such a solution in this paper.

### 3.2 Proposed technique

We propose a Galerkin technique for solving (4) numerically. A general Galerkin method for the integral equation is as follows. Let $V_n$ be a finite-dimensional function space with a basis set $\{\phi_i\}_{i=1}^{n}$, that is a subset of the (Hilbert) function space containing the solutions of (4). Then, we can approximate any solution $f$ as a linear combination of these basis functions:

$$f(\mathbf{x}) \approx f_n = \sum_{i=1}^{n} d_i \phi_i(\mathbf{x}) \quad (7)$$

Here, we have dropped the subscript $j$ and implicitly recognize that (7) and all following arguments hold for every eigenpair. Here, the subscript $n$ indicates that we are using an expansion in $n$ basis functions. If we substitute an approximate solution $f_n$ in (4), the two sides of the equation will not match exactly, resulting in a *residual*: the difference between the two sides, given by

$$r_n(\mathbf{x}) = \int_D K(\mathbf{x},\mathbf{y}) f_n(\mathbf{y}) d\mathbf{y} - \lambda_n f_n(\mathbf{x}) \quad (8)$$

Substituting (7) in (8)

$$r_n(\mathbf{x}) = \sum_{i=1}^{n} d_i \left\{ \int_D K(\mathbf{x},\mathbf{y})\phi_i(\mathbf{y}) d\mathbf{y} - \lambda_n \phi_i(\mathbf{x}) \right\} \quad (9)$$

where $\lambda_n$ and $d_i$ are all the unknowns that need to be computed, given the set $\phi_i$. The problem now is to estimate these unknowns so as to minimize the residual in some sense. The Galerkin criterion to accomplish this is to make the residual orthogonal to the basis:

$$\int_D r_n(\mathbf{x})\phi_k(\mathbf{x}) d\mathbf{x} = 0, \quad k = 1,\ldots,n \quad (10)$$

This ensures that the basis functions are completely utilized to "explain" as much of the true solution as possible using this finite-dimensional space $V_n$, as a result of which the residual is orthogonal to $V_n$. Convergence of the Galerkin technique has been well studied and established for continuous and bounded kernels [3], criteria that are easily satisfied by realistic kernels [1][12][16]. We can further manipulate (10) into a computation-friendly matrix form. Substituting (9), we get $\forall k \in \{1,\ldots,n\}$,

$$\sum_{i=1}^{n} d_i \left\{ \int_D \int_D K(\mathbf{x},\mathbf{y})\phi_i(\mathbf{y})\phi_k(\mathbf{x}) d\mathbf{x} d\mathbf{y} - \lambda_n \int_D \phi_i(\mathbf{x})\phi_k(\mathbf{x}) d\mathbf{x} \right\} = 0 \quad (11)$$

Writing

$$\begin{aligned} \mathbf{K}_{ik} &= \textstyle\int_D \int_D K(\mathbf{x},\mathbf{y})\phi_i(\mathbf{y})\phi_k(\mathbf{x}) d\mathbf{x} d\mathbf{y} \quad (12) \\ \mathbf{\Phi}_{ik} &= \textstyle\int_D \phi_i(\mathbf{x})\phi_k(\mathbf{x}) d\mathbf{x}, \quad \mathbf{d}_i = d_i, \end{aligned}$$

we get from (11)

$$\mathbf{K}\mathbf{d} = \lambda_n \mathbf{\Phi}\mathbf{d} \quad (13)$$

where the unknowns are $\lambda_n$ and the vector $\mathbf{d}$. This is the well-known *Generalized Eigenvalue Problem* (GEP), $\lambda_n$ being the eigenvalue and $\mathbf{d}$ being the eigenvector. We remind the reader that the $j$-th largest $\lambda_n$ and its corresponding eigenvector $d$ approximate the $j$-th eigenpair of (4). Further, if the basis set $\{\phi_i\}$ is orthogonal,

$$\int_D \phi_i(\mathbf{x})\phi_k(\mathbf{x}) d\mathbf{x} = 0, i \neq k, \quad (14)$$

then $\mathbf{\Phi}$ is a non-singular diagonal matrix. Hence, $\mathbf{\Phi}^{-1}$ is easily computable and we can simplify (13):

$$\begin{aligned} \mathbf{K}\mathbf{d} = \lambda_n \mathbf{\Phi}\mathbf{d} &\Rightarrow (\mathbf{\Phi}^{-1}\mathbf{K})\mathbf{d} = \lambda_n \mathbf{d} \quad (15) \\ (\mathbf{\Phi}^{-1}\mathbf{K})_{ik} &= \mathbf{K}_{ik} \cdot (\mathbf{\Phi}_{ii})^{-1} \quad (16) \end{aligned}$$

resulting in a standard eigenvalue problem (EP).

The development till now leaves us with three remaining steps: 1) determine the basis set $\{\phi_i\}$, 2) evaluate the integrals in (12), and 3) solve the GEP in (13) or the EP in (15). Step 3 being popular knowledge in the EDA community and easily solvable [17], we now focus on the first two.

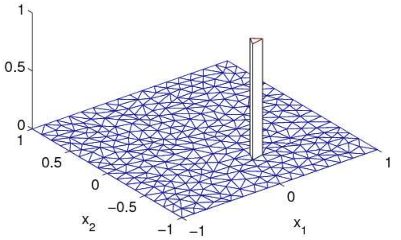


Figure 2: Triangular partition of chip area $D$ and one basis function.

# 4 Numerical Techniques

## 4.1 Galerkin expansion basis set

We choose a basis set $\phi_i$ of piecewise constant functions over a triangulation [2] of the chip area $D$:

$$D = \cup_{i=1}^{n} \triangle_i, \quad \phi_i(\mathbf{x}) = \begin{cases} 1, & \mathbf{x} \in \triangle_i \\ 0, & \mathbf{x} \notin \triangle_i \end{cases} \tag{17}$$

where $\triangle_i$ are triangles with a maximum overlap of one side. It is obvious that these functions are orthonormal. Fig. 2 shows an example triangulation and one such basis function. With such basis functions, that are zero everywhere outside one specific triangle, we can write the integrals in (12) as

$$\mathbf{K}_{ik} = \int_{\triangle_k}\int_{\triangle_i} K(\mathbf{x},\mathbf{y})\,d\mathbf{x}\,d\mathbf{y} = \mathbf{K}_{ki}, \quad \Phi_{ik} = \delta_{ik} a_i \tag{18}$$

where $\delta_{ik}$ is the Kronecker delta and $a_i$ is the area of $\triangle_i$.

## 4.2 Numerical integration

We still need to evaluate the integral in (18): we now propose a simple numerical technique that exploits the triangulation, and study its convergence. Consider the integral

$$I = \int_{\triangle} g(\mathbf{x})\,d\mathbf{x} \tag{19}$$

over a generic triangle element denoted by $\triangle$. We approximate $g(\mathbf{x})$ by its value at the centroid $\mathbf{x}_\triangle$ of $\triangle$. Then we can write

$$I \approx \hat{I} = \int_{\triangle} g(\mathbf{x}_\triangle)\,d\mathbf{x} = g(\mathbf{x}_\triangle) a_\triangle \tag{20}$$

Hence, the double integral in (18) is approximated by

$$\mathbf{K}_{ik} \approx \int_{\triangle_k} K(\mathbf{x}_{\triangle_i},\mathbf{y}) a_i\,d\mathbf{y} \approx K(\mathbf{x}_{\triangle_i},\mathbf{x}_{\triangle_k}) a_i a_k \approx \mathbf{K}_{ki} \tag{21}$$

The integration error is given by

$$E = I - \hat{I} = \int_{\triangle} (g(\mathbf{x}) - g(\mathbf{x}_\triangle))\,d\mathbf{x} \tag{22}$$

The Taylor's expansion of $g$ around $\mathbf{x}_\triangle$ can be written as [18]

$$g(\mathbf{x} = \mathbf{x}_\triangle + \delta\mathbf{x}) = g(\mathbf{x}_\triangle) + \sum_{i=1}^{2} \delta x_i \int_0^1 \frac{\partial g}{\partial x_i}(\mathbf{x}_\triangle + t\delta\mathbf{x})\,dt \tag{23}$$

Then, the error can be written as

$$E = \int_{\triangle} \left\{ \sum_{i=1}^{2} \delta x_i \int_0^1 \frac{\partial g}{\partial x_i}(\mathbf{x}_\triangle + t\delta\mathbf{x})\,dt \right\} d\mathbf{x} \tag{24}$$

If we define $h$ as the maximum triangle side in the partition of $D$, then $\delta x_i$ is never more than $h$ while integrating over any triangle. Also, assume that the first derivative of $g$ is bounded over $D$:

$$\left| \frac{\partial g(\mathbf{x})}{\partial x_i} \right| \leq M_i, \forall \mathbf{x} \in D \tag{25}$$

for some finite, non-negative $M_i$. Then, from (24)

$$|E| \leq \int_{\triangle} \left\{ \sum_{i=1}^{2} h \int_0^1 M_i\,dt \right\} d\mathbf{x} = h a_\triangle (M_1 + M_2) \tag{26}$$

which is linearly decreasing with $h$. Using this bound in (21), we can easily show that the double integral approximation error also decreases linearly with $h$. Hence, we have proved the following theorem.

**Theorem 2** *Let $D$ be a polygonl region in the plane $\mathbf{R}^2$, $\{\triangle_i\}_1^n$ be a triangulation of $D$ and $h$ be the maximum triangle side. Then, if $K$ has well-defined, bounded first derivatives over $D$, $\forall i,k \in \{1,\ldots,n\}$*

$$\lim_{h\to 0} \left| \int_{\triangle_k}\int_{\triangle_i} K(\mathbf{x},\mathbf{y})\,d\mathbf{x}\,d\mathbf{y} - K(\mathbf{x}_{\triangle_i},\mathbf{x}_{\triangle_k}) a_i a_k \right| = 0 \tag{27}$$

*where $\mathbf{x}_{\triangle_i}$ indicates the centroid of $\triangle_i$ and $a_i$ its area, and the convergence is linear in $h$.*

In other words, the integration error tends to zero as we increase the number of triangles, $n$, establishing the validity of this numerical integration technique.

We note that we are using two levels of approximation here: 1) a finite representation of the eigenfunctions in (7), where the $\phi_i$ are defined by (17), and 2) a numerical approximation of the double integral in (18) using a constant, as in (21). [3] establishes a linear rate of convergence of the Galerkin method using such approximation, and hence, for our complete technique as described in this paper. Higher order piecewise polynomials can also be used as the basis set, along with high order numerical integration. These high order techniques would result in more accurate estimates of the eigenpairs, and there are no restrictions on their use in this setting. However, our simpler technique shows acceptable accuracy, as demonstrated in the results section.

## 4.3 Reconstructing the stochastic process

Equation (3) suggests that we can use a few RVs $\xi_i$ to approximately construct the entire stochastic process $\mathbf{p}$ using a linear transform. If we use the first $r$ ($r \leq n$) KLE eigenpairs to approximate the stochastic process, we can define the matrices $\Lambda_r$ ($r \times r$) and $\mathbf{D}_r$ ($n \times r$): $\Lambda_r$ is the diagonal matrix containing the KLE eigenvalues, and the $i$-th column of $\mathbf{D}_r$ is the eigenvector of (13) corresponding to the $i$-th eigenvalue in $\Lambda_r$. Then, we define the matrix $\mathbf{D}_\lambda = \mathbf{D}_r\sqrt{\Lambda_r}$, where the square-root is taken element-wise. Now, if we generate a random sample $\bar{\xi}$ in the reduced $r$-dimensional space of RVs $\xi_i$, we can use (3), (7) and (17) to linearly transform it to the corresponding values of the relevant statistical parameter over the entire chip. We can write this as

$$\mathbf{p}_\triangle = \mathbf{D}_\lambda \bar{\xi} \tag{28}$$

where the $i$-th element $p_{\triangle_i}$, of the vector $\mathbf{p}_\triangle$, approximates the value of $\mathbf{p}(\mathbf{x},\theta)$ in $\triangle_i$ as a constant over $\triangle_i$.

# 5 Experiments

## 5.1 Experimental setup

We stress that the grid-model model is a general, *algorithm independent* technique to model chip-wide intra-die

[2] Although any meshing is usable, triangulation makes it easy to select the number of mesh elements and constrain their shape, using widely available tools [24]. This is in contrast to, say, uniform rectangles, which require quadratically more elements with every increase in resolution.

**Algorithm 1** Generate $N$ samples for Monte Carlo STA

```
for all stat. parameters $p_j$ do
    $\mathbf{K}_j \leftarrow$ CovMatrix($K_j$, $\{\mathbf{g}_i\}$)
    $\mathbf{U}_j \leftarrow$ CholeskyUpperFactor($\mathbf{K}_j$)
    $\mathbf{P}_j \leftarrow$ RandNormal($N$, $N_p$) $\cdot \mathbf{U}_j$
end for
```

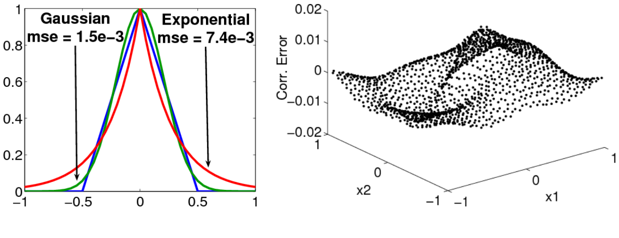
**Algorithm 2** Generate $N$ samples for covariance kernel STA

```
for all stat. parameters $p_j$ do
    $\Xi_j \leftarrow$ RandNormal($N$, $r$)
    $\mathbf{P}_{j\triangle} \leftarrow \mathbf{D}_\lambda \Xi_j$
    for $i = 1$ to $N_g$ do
        $t \leftarrow$ IndexOfContainingTriangle($\mathbf{g}_i$)
        Row($i$, $\mathbf{P}_j$) $\leftarrow$ Row($t$, $\mathbf{P}_{j\triangle}$)
    end for
end for
```

variations. Nevertheless, we need a concrete algorithm to demonstrate its merits. The obvious choice is SSTA, since timing is highly sensitive to such variations. We use as a reference a gate-level Monte Carlo (MC)-based SSTA. This offers two useful virtues. First, it frees the experiment from any model-specific errors that may introduce noise in the comparison (e.g., due to linear models in [6], polynomial chaos models in [2], grid correlation models in [5]). Second, it immediately shows us the impact of reducing the number of random variables from several-per-gate to few tens (e.g. 25).

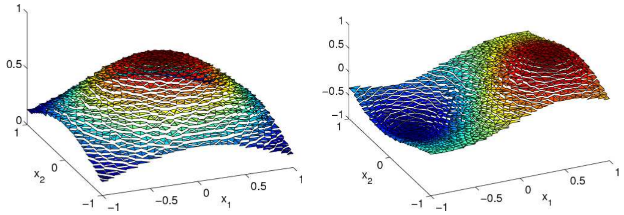
The reference MC algorithm we use is very simple. Assume $N_p$ statistical parameters (each representing a stochastic process with covariance kernel $K_j$) $N_g$ gates on the chip, and $N$ MC samples. First use Algorithm 1 to generate $N_p$ matrices, $\{\mathbf{P}_j\}_1^{N_p}$, of size $N \times N_g$, each containing $N$ points that follow the corresponding $K_j$. Note that $\mathbf{g}_i$ is the location of the $i$-th gate. For the $i$-th STA run in the MC simulation, use the gate parameter values from the $i$-th row of each matrix $\mathbf{P}_j$. Note that $\mathbf{P}_j$ are mutually independent. Our KLE-based technique is also used in the same MC framework, the difference being in the generation of the gate parameter samples: Algorithm 2 is used instead of Algorithm 1. Note that each column of the intermediate matrix $\Xi_j$ is one random sample in $r$-dimensional space. IndexOfContainingTriangle() can be made efficient using some space indexing (grid, tree, etc.) scheme: the details are skipped due to space constraints.

We use a Gaussian kernel ($K(\mathbf{x},\mathbf{y}) = \exp(-c||\mathbf{x}-\mathbf{y}||_2^2)$, Fig. 1(a)) for our tests. Using measurement data, [12] suggests a near linear isotropic kernel (assuming normalized chip sides). The isotropic linear kernel, however, can be invalid [1]. [16] suggests using an isotropic exponential kernel. However, the Gaussian kernel fits the measurement data supported linear kernel better than the exponential kernel, as shown in Fig. 3(a) by best fits to the linear kernel in 1-D. We compute $c$ to best fit an isotropic linear kernel in 2-D with correlation distance equal to half the normalized chip length (a cone with a base radius of half chip length).



(a) (b)

Figure 3: (a) Best fit of Gaussian and Exponential kernels to the linear kernel suggested in [12]. (b) Error in reconstruction of 2-D Gaussian kernel from 25 eigenpairs computed using our technique.



(a) (b)

Figure 4: Gaussian kernel eigenfunction: (a) first, (b) second.

We now describe the structure and modeling assumptions used in our STA algorithm (the core timer inside the Monte Carlo loops) and gate library. The STA computes signal delays at all the circuit nodes, using the Elmore delay metric [19] for wire delay, the PERI technique [20] with the Bakoglu metric [21] for wire slew, and rank-one quadratic functions [22] to model gate delay and gate output slew. The gate output slew and gate delay are modeled as functions of the input slew and 4 statistical parameters: $L$, $W$, $V_t$ and $t_{ox}$. Note that there is no restriction imposed by our technique on the type of gate model used. We use standard logic netlist benchmarks from the ISCAS85 and ISCAS89 sets. All the test circuits were placed using the Capo placer [23] (MetaPlacer), and half-perimeter wirelength was used to model the wire loads. The gates and wires were implemented using the 90nm Cadence Generic PDK. All statistical parameters are assumed to have Gaussian distributions.

### 5.2 Experimental results

We first try to reconstruct the Gaussian kernel (Fig. 1(a)) using only $r = 25$ numerically computed eigenpairs. The triangular mesh was generated using Triangle [24] with minimum angle of 28° and maximum triangle area of 0.1% of the chip area, resulting in $n = 1546$ triangles. Fig. 3(b) shows the error in the reconstruction of the kernel: the maximum error magnitude is a small 0.016. Fig. 4 shows the first two eigenfunctions, where we can see the Fourier series type behavior: higher eigenfunctions model the higher frequencies in the correlation. The eigenvalues decay very rapidly (Fig. 5). We have chosen $r$ such that $\lambda_{200}(n-200) + \sum_{i=r+1}^{200} \lambda_i \leq 0.01 \sum_{i=1}^{r} \lambda_i$. The left side of the inequality is an upper bound on the sum of all unused $n-r$ eigenvalues, given that we have computed only the first 200. This bound is less than 1% of the sum of the first $r$ eigenvalues, giving us $r = 25$. This criterion tries to ensure that most of the variation across the chip is accounted for by the chosen eigenpairs. Let $\sigma_d$ be the standard deviation of delay at any circuit node, and let us consider a single circuit, c1908 (ISCAS85, 880 gates). Now, we take a 100K-sample Monte Carlo STA run as the reference, and look at the relative error in the estimate of $\sigma_d$ computed using our covariance kernel-based STA with a varying parameter. We consider two parameters: 1) increasing number of eigenpairs $r$, and 2) increasing number of triangles $n$. The error is averaged across all the outputs of the circuit. The results are shown in Fig. 6. We see expected behavior: accuracy increases with increasing $r$ and $n$. Note that the error has some noise because the reference result

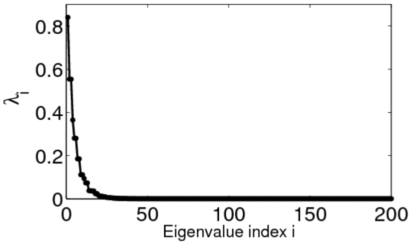
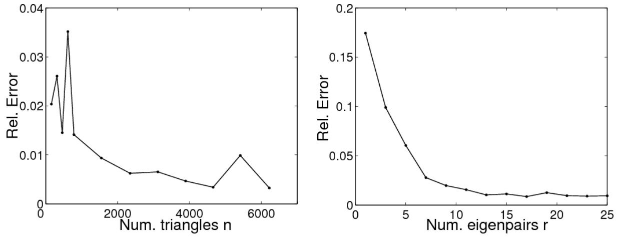


Figure 5: Gaussian kernel: the first 200 eigenvalues



(a) (b)

Figure 6: Error in the covariance kernel-based STA estimate of std. dev. of delay for increasing (a) number of eigenpairs $r$ ($n = 1546$), and (b) number of triangles $n$ ($r = 25$).

(Monte Carlo STA) is approximate and random; but the general trend still holds. The error in the delay mean is much smaller (0.025% for 25 eigenpairs and 1546 triangles) and shows similar behavior. We use $(r,n) = (25,1546)$ for subsequent tests. This combination has error $< 1\%$. Finally, we compare the mean and standard deviation of the circuit delay computed by Monte Carlo STA and our covariance kernel-based STA, using 100K samples each. Table 1 shows the mismatch between the estimates as a percentage of the estimate from Monte Carlo STA; the mismatch is within 5.7%. The algorithms were implemented in C++ on a 2.8 GHz dual core Opteron. Note that the number of gates $N_g$ is the number of (correlated) RVs handled by the Monte Carlo STA, for each statistical parameter (e.g. $t_{ox}$): this is reduced to 25 for KLE. Of course, KLE comes with the overhead of the reconstruction in (28). However, we start seeing the speed gains of the reduced dimensionality very soon, as shown by Table 1. We expect these trends to replicate in other CAD algorithms where the complexity increases with dimensionality. Further, eigenpair computation takes 11.2s, using Matlab.

| Circuit | $N_g$ (gates) | $e_\mu$ (%) | $e_\sigma$ (%) | Speedup |
|---|---|---|---|---|
| c880 | 383 | 0.020 | 0.593 | 0.29 |
| c1355 | 546 | 0.057 | 1.711 | 0.41 |
| c1908 | 880 | 0.025 | 1.026 | 0.61 |
| c3540 | 1669 | 0.003 | 1.288 | 1.25 |
| c5315 | 2307 | 0.015 | 0.033 | 1.79 |
| c6288 | 2416 | 0.042 | 0.062 | 2.07 |
| s5378 | 2779 | 0.058 | 1.534 | 2.56 |
| c7552 | 3512 | 0.031 | 0.768 | 3.43 |
| s9234 | 5597 | 0.013 | 2.635 | 5.81 |
| s13207 | 7951 | 0.034 | 2.448 | 8.36 |
| s15850 | 9772 | 0.038 | 1.394 | 10.57 |
| s35932 | 16065 | 0.029 | 5.647 | 2.22 |
| s38584 | 19253 | 0.109 | 2.755 | 10.65 |
| s38417 | 22179 | 0.020 | 1.108 | 3.77 |

Table 1: Percentage mismatch in worst delay mean ($e_\mu$) and std. dev. ($e_\sigma$) between Monte Carlo STA and our covariance kernel-based STA (100K samples).

# 6 Conclusions

Spatial correlation in random intra-die manufacturing variation has typically been modeled using simple grid-based models, that are often impractical and lack rigorous construction. The recently proposed grid-less random field model overcomes these problems, but no methods have been suggested for their effective use with generic kernels in CAD tools. In this paper we have proposed a robust and efficient numerical Galerkin method that exploits Karhunen Loéve Expansion to approximate the random field using only a few (e.g. 25) RVs, with acceptable loss of accuracy. We also establish convergence properties of the method. For a simple Monte Carlo-based STA algorithm, the implementation is straightforward, and already shows speedups up to 10x.

**Acknowledgments** This work was funded by the MARCO Focus Center for Circuit and System Solutions (C2S2) and the Semiconductor Research Corporation.

# References

[1] J. Xiong et al, "Robust extraction of spatial correlation," *IEEE Trans. CAD*, 26(4), Apr, 2007.

[2] S. Bhardwaj et al, "A framework for statistical timing analysis using non-linear delay and slew models," *ICCAD*, 2006.

[3] K. E. Atkinson, "The numerical solution of integral equations of the second kind," Cambridge Univ. Press, 1997.

[4] M. Loéve, "Probability Theory I & II", Springer, 4 ed., 1977.

[5] H. Chang and S. Sapatnekar, "Statistical timing analysis under spatial correlations," *DAC*, 2005.

[6] C. Visweswariah, et al, "First-order incremental block-based statistical timing analysis," *DAC*, 2004.

[7] I. A. Ferzli and F. N. Najm, "Analysis and verification of power grids considering process-induced leakage-current variations," *IEEE Trans. CAD*, 25(1), Jan, 2006.

[8] R. G. Ghanem and P. D. Spanos, "Stochastic finite elements: a spectral approach," Springer-Verlag, 1991.

[9] P. Gupta and F.-L. Heng, "Toward a systematic-variation aware timing methodology," *DAC*, 2004.

[10] M. Hane et al, "Atomistic 3D process/device simulation considering gate line-edge roughness and poly-Si random crystal orientation effects," *IEDM*, 2003.

[11] M. J. M. Pelgrom et al, "Matching properties of MOS transistors," *IEEE JSSC*, 24(5), Oct, 1989.

[12] P. Friedberg et al, "Modeling within-die spatial correlation effects for process-design co-optimization,", *ISQED*, 2005.

[13] A. Agarwal et al, "Statistical delay computation considering spatial correlations," *ASPDAC*, 2003.

[14] V. Khandelwal et al, "A general framework for accurate statistical timing analysis considering correlations," *DAC*, 2005.

[15] B. J. Lee et al, "Refined statistical static timing analysis through learning spatial delay correlations," *DAC*, 2006.

[16] F. Liu, "A general framework for spatial correlation modeling in VLSI design," *DAC*, 2007.

[17] G. H. Golub and C. F. Van Loan,"Matrix Computations," JHU Press, 3 ed., 1996.

[18] J. T. Day, "On the convergence of Taylor series for functions of n variables," Math. Maga., 40(5), Nov, 1967.

[19] J. Rubenstein et al, "Signal delay in RC tree networks," *IEEE Trans. CAD*, CAD-2, Jul, 1983.

[20] C. V. Kashyap et al, "Closed-form expressions for extending step delay and slew metrics to ramp inputs for RC trees," *IEEE Trans. CAD*, 23(4), Apr, 2004.

[21] H. B. Bakoglu, "Circuits, interconnections, and packaging for VLSI," Addison-Wesley, 1990.

[22] X. Li et al, "Projection-based performance modeling for inter/intra-die variations," *ICCAD*, 2005.

[23] A. E. Caldwell et al, "Can recursive bisection alone produce routable placements?," *DAC*, 2000.

[24] J. R. Shewchuk, "Delaunay refinement algorithms for triangular mesh generation," *Comp. Geom.: Theory and Appls.*, 22(1-3), May, 2002.