Enabling High-Dimensional Hierarchical Uncertainty Quantification by ANOVA and Tensor-Train Decomposition

Citation: Zheng Zhang, Xiu Yang, Ivan V. Oseledets, George E. Karniadakis, and Luca Daniel. "Enabling High-Dimensional Hierarchical Uncertainty Quantification by ANOVA and Tensor-Train Decomposition." IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 34, no. 1 (January 2015): 63–76.

As Published: http://dx.doi.org/10.1109/TCAD.2014.2369505

Publisher: Institute of Electrical and Electronics Engineers (IEEE)

Persistent URL: http://hdl.handle.net/1721.1/99952

Version: Author's final manuscript: final author's manuscript post peer review, without publisher's formatting or copy editing

Enabling High-Dimensional Hierarchical Uncertainty Quantification by ANOVA and Tensor-Train Decomposition

Zheng Zhang, Xiu Yang, Ivan V. Oseledets, George Em Karniadakis, and Luca Daniel
Accepted by IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems

***Abstract*—Hierarchical uncertainty quantification can reduce the computational cost of stochastic circuit simulation by employing spectral methods at different levels. This paper presents an efficient framework to simulate hierarchically some challenging stochastic circuits/systems that include high-dimensional subsystems. Due to the high parameter dimensionality, it is challenging to both extract surrogate models at the low level of the design hierarchy and to handle them in the high-level simulation. In this paper, we develop an efficient ANOVA-based stochastic circuit/MEMS simulator to extract efficiently the surrogate models at the low level. In order to avoid the curse of dimensionality, we employ tensor-train decomposition at the high level to construct the basis functions and Gauss quadrature points. As a demonstration, we verify our algorithm on a stochastic oscillator with four MEMS capacitors and 184 random parameters. This challenging example is simulated efficiently by our simulator at the cost of only 10 minutes in MATLAB on a regular personal computer.**

***Index Terms*—Uncertainty quantification, hierarchical uncertainty quantification, generalized polynomial chaos, stochastic modeling and simulation, circuit simulation, MEMS simulation, high dimensionality, analysis of variance (ANOVA), tensor train.**

I. Introduction

PROCESS variations have become a major concern in submicron and nano-scale chip design [2]–[6]. In order to improve chip performances, it is highly desirable to develop efficient stochastic simulators to quantify the uncertainties of integrated circuits and microelectromechanical systems (MEMS). Recently, stochastic spectral methods [7]–[12] have emerged as a promising alternative to Monte Carlo techniques [13]. The key idea is to represent the stochastic solution as a linear combination of some basis functions (e.g., generalized polynomial chaos [8]), and then compute the solution by stochastic Galerkin [7], stochastic collocation [9]–[12] or stochastic testing [14]–[16] methods. Due to the fast convergence rate, such techniques have been successfully applied in the stochastic analysis of integrated circuits [14]–[20], VLSI interconnects [21]–[25], electromagnetic [26] and MEMS devices [1], [27], achieving significant speedup over Monte Carlo when the parameter dimensionality is small or medium.

Since many electronic systems are designed in a hierarchical way, it is possible to exploit such structure and simulate a complex circuit by hierarchical uncertainty quantification [28][1]. Specifically, one can first utilize stochastic spectral methods to extract surrogate models for each block. Then, circuit equations describing the interconnection of blocks may be solved with stochastic spectral methods by treating each block as a single random parameter. Typical application examples include (but are not limited to) analog/mixed-signal systems (e.g., phase-lock loops) and MEMS/IC co-design. In our preliminary conference paper [1], this method was employed to simulate a low-dimensional stochastic oscillator with 9 random parameters, achieving $250\times$ speedup over the hierarchical Monte-Carlo method proposed in [32].

Paper Contributions. This paper extends the recently developed hierarchical uncertainty quantification method [28] to the challenging cases that include subsystems with high dimensionality (i.e., with a large number of parameters). Due to such high dimensionality, it is too expensive to extract a surrogate model for each subsystem by any standard stochastic spectral method. It is also non-trivial to perform high-level simulation with a stochastic spectral method, due to the high-dimensional integration involved when computing the basis functions and Gauss quadrature rules for each subsystem. In order to reduce the computational cost, this work develops some fast numerical algorithms to accelerate the simulations at both levels:

- At the low level, we develop a sparse stochastic testing simulator based on adaptive anchored ANOVA [33]–[37] to efficiently simulate each subsystem. This approach exploits the sparsity on-the-fly, and it turns out to be suitable for many circuit and MEMS problems. This algorithm was reported in our preliminary conference paper [1] and was used for the global sensitivity analysis of analog integrated circuits.

Some preliminary results of this work have been reported in [1]. This work was funded by the MIT-SkolTech program. I. Oseledets was also supported by the Russian Science Foundation under Grant 14-11-00659.

Z. Zhang and L. Daniel are with the Research Laboratory of Electronics, Massachusetts Institute of Technology (MIT), Cambridge, MA 02139, USA (e-mail: z_zhang@mit.edu, luca@mit.edu).

X. Yang was with the Division of Applied Mathematics, Brown University, Providence, RI 02912. Now he is with the Pacific Northwest National Laboratory, Richland, WA 99352, USA (e-mail: xiu.yang@pnnl.gov).

G. Karniadakis is with the Division of Applied Mathematics, Brown University, Providence, RI 02912, USA (e-mail: george_karniadakis@brown.edu).

Ivan V. Oseledets is with the Skolkovo Institute of Science and Technology, Skolkovo 143025, Russia (e-mail: ivan.oseledets@gmail.com).

[1]Design hierarchy can be found in many engineering fields. In the recent work [29] a hierarchical stochastic analysis and optimization framework based on multi-fidelity models [30], [31] was proposed for aircraft design.

- In the high-level stochastic simulation, we accelerate the three-term recurrence relation [38] by tensor-train decomposition [39]–[41]. Our algorithm has a linear complexity with respect to the parameter dimensionality, generating a set of basis functions and Gauss quadrature points with high accuracy (close to the machine precision). This algorithm was not reported in [1].

II. Background Review

This section first reviews the recently developed stochastic testing circuit/MEMS simulator [14]–[16] and hierarchical uncertainty quantification [28]. Then we introduce some background about tensor and tensor decomposition.

A. Stochastic Testing Circuit/MEMS Simulator

Given a circuit netlist (or a MEMS 3D schematic file), device models and process variation descriptions, one can set up a stochastic differential algebraic equation:

$$\frac{d\vec{q}\left(\vec{x}(t,\vec{\xi}),\vec{\xi}\right)}{dt}+\vec{f}\left(\vec{x}(t,\vec{\xi}),\vec{\xi},u(t)\right)=0 \tag{1}$$

where $\vec{u}(t)$ is the input signal, $\vec{\xi}$=$[\xi_1,\cdots,\xi_d]\in\Omega\subseteq\mathbb{R}^d$ are d mutually independent random variables describing process variations. The joint probability density function of $\vec{\xi}$ is

$$\rho(\vec{\xi})=\prod_{k=1}^{d}\rho_k\left(\xi_k\right), \tag{2}$$

where $\rho_k\left(\xi_k\right)$ is the marginal probability density function of $\xi_k\in\Omega_k$. In circuit analysis, $\vec{x}\in\mathbb{R}^n$ denotes nodal voltages and branch currents; $\vec{q}\in\mathbb{R}^n$ and $\vec{f}\in\mathbb{R}^n$ represent charge/flux and current/voltage, respectively. In MEMS analysis, Eq. (1) is the equivalent form of a commonly used 2nd-order differential equation [1], [42]; $\vec{x}$ includes displacements, rotations and their first-order derivatives with respect to time t.

When $\vec{x}(\vec{\xi},t)$ has a bounded variance and smoothly depends on $\vec{\xi}$, we can approximate it by a truncated generalized polynomial chaos expansion [8]

$$\vec{x}(t,\vec{\xi})\approx\tilde{x}(t,\vec{\xi})=\sum_{\vec{\alpha}\in\mathcal{P}}\hat{x}_{\vec{\alpha}}(t)H_{\vec{\alpha}}(\vec{\xi}) \tag{3}$$

where $\hat{x}_{\vec{\alpha}}(t)\in\mathbb{R}^n$ denotes a coefficient indexed by vector $\vec{\alpha}=[\alpha_1,\cdots,\alpha_d]\in\mathbb{N}^d$, and the basis function $H_{\vec{\alpha}}(\vec{\xi})$ is a multivariate polynomial with the highest order of ξ_i being α_i. In practical implementations, it is popular to set the highest total degree of the polynomials as p, then $\mathcal{P}=\{\vec{\alpha}|\ \alpha_k\in\mathbb{N},\ 0\le\alpha_1+\cdots+\alpha_d\le p\}$ and the total number of basis functions is

$$K=\left(\begin{array}{c}p+d\\p\end{array}\right)=\frac{(p+d)!}{p!d!}. \tag{4}$$

For any integer j in $[1,K]$, there is a one-to-one correspondence between j and $\vec{\alpha}$. As a result, we can denote a basis function as $H_j(\vec{\xi})$ and rewrite (3) as

$$\vec{x}(t,\vec{\xi})\approx\tilde{x}(t,\vec{\xi})=\sum_{j=1}^{K}\hat{x}_j(t)H_j(\vec{\xi}). \tag{5}$$

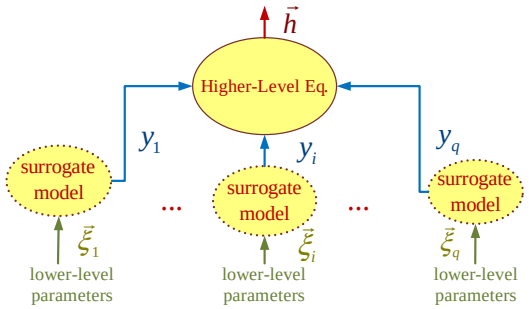


Fig. 1. Demonstration of hierarchical uncertainty quantification.

In order to compute $\vec{x}(t,\vec{\xi})$, stochastic testing [14]–[16] substitutes $\tilde{x}(t,\vec{\xi})$ into (1) and forces the residual to zero at K testing samples of $\vec{\xi}$. This gives a deterministic differential algebraic equation of size nK

$$\frac{d\mathbf{q}(\hat{\mathbf{x}}(t))}{dt}+\mathbf{f}\left(\hat{\mathbf{x}}(t),u(t)\right)=0, \tag{6}$$

where the state vector $\hat{\mathbf{x}}(t)$ contains all coefficients in (3). Stochastic testing then solves Eq. (6) with a linear complexity of K and with adaptive time stepping, and it has shown higher efficiency than standard stochastic Galerkin and stochastic collocation methods in circuit simulation [14], [15].

1) Constructing Basis Functions: The basis function $H_{\vec{\alpha}}(\vec{\xi})$ is constructed as follows (see Section II of [16] for details):

- First, for ξ_i one constructs a set of degree-α_i orthonormal univariate polynomials $\left\{\varphi^i_{\alpha_i}(\xi_i)\right\}_{\alpha_i=0}^{p}$ according to its marginal probability density $\rho_i(\xi_i)$.
- Next, based on the obtained univariate polynomials of each random parameter one constructs the multivariate basis function: $H_{\vec{\alpha}}(\vec{\xi})$=$\prod_{i=1}^{d}\varphi^i_{\alpha_i}(\xi_i)$.

The obtained basis functions are orthonormal polynomials in the multi-dimensional parameter space Ω with the density measure $\rho(\vec{\xi})$. As a result, some statistical information can be easily obtained. For example, the mean value and variance of $\vec{x}(t,\vec{\xi})$ are $\hat{x}_0(t)$ and $\sum_{\vec{\alpha}\neq0}\left(\hat{x}_{\vec{\alpha}}(t)\right)^2$, respectively.

2) Testing Point Selection: The selection of testing points influence the numerical accuracy of the simulator. In stochastic testing, the testing points $\{\vec{\xi}^j\}_{j=1}^{K}$ are selected by the following two steps (see Section III-C of [14] for details):

- First, compute a set of multi-dimensional quadrature points. Such quadrature points should give accurate results for evaluating the numerical integration of any multivariate polynomial of $\vec{\xi}$ over Ω [with density measure $\rho(\vec{\xi})$] when the polynomial degree is $\le 2p$.
- Next, among the obtained quadrature points, we select the K samples with the largest quadrature weights under the constraint that $\mathbf{V}\in\mathbb{R}^{K\times K}$ is well-conditioned. The (j,k) element of $\mathbf{V}$ is $H_k(\vec{\xi}^j)$.

B. Hierarchical Uncertainty Quantification

Consider Fig. 1, where an electronic system has q subsystems. The output y_i of a subsystem is influenced by some process variations $\vec{\xi}_i\in\mathbb{R}^{d_i}$, and the output $\vec{h}$ of the

whole system depends on all random parameters $\vec{\xi}_i$'s. For simplicity, in this paper we assume that y_i only depends on $\vec{\xi}_i$ and does not change with time or frequency. Directly simulating the whole system can be expensive due to the large problem size and high parameter dimensionality. If y_i's are mutually independent and smoothly dependent on $\vec{\xi}_i$'s, we can accelerate the simulation in a hierarchical way [28]:

- First, perform low-level uncertainty quantification. We use stochastic testing to simulate each block, obtaining a generalized polynomial expansion for each y_i. In this step we can also employ other stochastic spectral methods such as stochastic Galerkin or stochastic collocation.
- Next, perform high-Level uncertainty quantification. By treating y_i's as the inputs of the high-level equation, we use stochastic testing again to efficiently compute $\vec{h}$. Since y_i has been assumed independent of time and frequency, we can treat it as a random parameter.

In order to apply stochastic spectral methods at the high level, we need to compute a set of *specialized* orthonormal polynomials and Gauss quadrature points/weights for each input random parameter. For the sake of numerical stability, we define a zero-mean unit-variance random variable ζ_i for each subsystem, by shifting and scaling y_i. The intermediate variables $\vec{\zeta} = [\zeta_1, \cdots, \zeta_q]$ are used as the random parameters in the high-level equation. Dropping the subscript for simplicity, we denote a general intermediate-level random parameter by ζ and its probability density function by $\rho(\zeta)$ (which is actually unknown), then we can construct $p+1$ orthogonal polynomials $\{\pi_j(\zeta)\}_{j=0}^{p}$ via a three-term recurrence relation [38]

$$\begin{aligned} &\pi_{j+1}(\zeta) = (\zeta - \gamma_j)\,\pi_j(\zeta) - \kappa_j \pi_{j-1}(\zeta), \\ &\pi_{-1}(\zeta) = 0, \;\; \pi_0(\zeta) = 1, \;\; j = 0, \cdots, p-1 \end{aligned} \tag{7}$$

with

$$\gamma_j = \frac{\int_{\mathbb{R}} \zeta \pi_j^2(\zeta)\rho(\zeta)d\zeta}{\int_{\mathbb{R}} \pi_j^2(\zeta)\rho(\zeta)d\zeta}, \;\; \kappa_{j+1} = \frac{\int_{\mathbb{R}} \pi_{j+1}^2(\zeta)\rho(\zeta)d\zeta}{\int_{\mathbb{R}} \pi_j^2(\zeta)\rho(\zeta)d\zeta} \tag{8}$$

and $\kappa_0 = 1$, where $\pi_j(\zeta)$ is a degree-j polynomial with a leading coefficient 1. The first $p+1$ univariate basis functions can be obtained by normalization:

$$\phi_j(\zeta) = \frac{\pi_j(\zeta)}{\sqrt{\kappa_0 \kappa_1 \cdots \kappa_j}}, \text{ for } j = 0, 1, \cdots, p. \tag{9}$$

The parameters κ_j's and γ_j's can be further used to form a symmetric tridiagonal matrix $\mathbf{J} \in \mathbb{R}^{(p+1)\times(p+1)}$:

$$\mathbf{J}(j,k) = \begin{cases} \gamma_{j-1}, \text{ if } j = k \\ \sqrt{\kappa_j}, \text{ if } k = j+1 \\ \sqrt{\kappa_k}, \text{ if } k = j-1 \\ 0, \text{ otherwise} \end{cases} \text{ for } 1 \le j,k \le p+1. \tag{10}$$

Let $\mathbf{J} = \mathbf{U}\Sigma\mathbf{U}^T$ be an eigenvalue decomposition, where $\mathbf{U}$ is a unitary matrix. The j-th quadrature point and weight of ζ are $\Sigma(j,j)$ and $(\mathbf{U}(1,j))^2$, respectively [43].

Challenges in High Dimension. When d_i is large, it is difficult to implement hierarchical uncertainty quantification. First, it is non-trivial to obtain a generalized polynomial chaos expansion for y_i, since a huge number of basis functions and samples are required to obtain y_i. Second, when high accuracy is required, it is expensive to implement (7) due to the non-trivial integrals when computing κ_j and γ_j. Since the density function of ζ_i is unknown, the integrals must be evaluated in the domain of $\vec{\xi}_i$, with a cost growing exponentially with d_i when a deterministic quadrature rule is used.

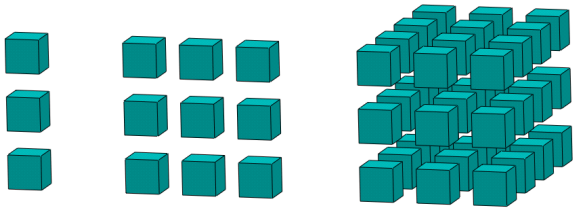


Fig. 2. Demonstration of a vector (left), a matrix (center) and a 3-mode tensor (right).

C. Tensor and Tensor Decomposition

Definition 1 (Tensor). *A tensor $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{N_1 \times N_2 \times \cdots \times N_d}$ is a multi-mode (or multi-way) data array. The mode (or way) is d, the number of dimensions. The size of the k-th dimension is N_k. An element of the tensor is $\boldsymbol{\mathcal{A}}(i_1, \cdots, i_d)$, where the positive integer i_k is the index for the k-th dimension and $1 \le i_k \le N_k$. The total number of elements of $\boldsymbol{\mathcal{A}}$ is $N_1 \times \cdots \times N_d$.*

As a demonstration, we have shown a vector (1-mode tensor) in $\mathbb{R}^{3\times 1}$, a matrix (2-mode tensor) in $\mathbb{R}^{3\times 3}$ and a 3-mode tensor in $\mathbb{R}^{3\times 3\times 4}$ in Fig. 2, where each small cube represents a scalar.

Definition 2 (Inner Product of Two Tensors). *For $\boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}} \in \mathbb{R}^{N_1 \times N_2 \times \cdots \times N_d}$, their inner product is defined as the sum of their element-wise product*

$$\langle \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}} \rangle = \sum_{i_1, \cdots, i_d} \boldsymbol{\mathcal{A}}(i_1, \cdots i_d)\, \boldsymbol{\mathcal{B}}(i_1, \cdots i_d). \tag{11}$$

Definition 3 (Frobenius Norm of A Tensor). *For $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{N_1 \times N_2 \times \cdots \times N_d}$, its Frobenius norm is defined as*

$$\|\boldsymbol{\mathcal{A}}\|_F = \sqrt{\langle \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{A}} \rangle}. \tag{12}$$

Definition 4 (Rank-One Tensors). *A d-mode tensor $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{N_1 \times \cdots \times N_d}$ is rank one if it can be written as the outer product of d vectors*

$$\boldsymbol{\mathcal{A}} = \mathbf{v}^{(1)} \circ \mathbf{v}^{(2)} \cdots \circ \mathbf{v}^{(d)}, \text{ with } \mathbf{v}^{(k)} \in \mathbb{R}^{N_k} \tag{13}$$

where $\circ$ denotes the outer product operation. This means that

$$\boldsymbol{\mathcal{A}}(i_1, \cdots, i_d) = \prod_{k=1}^{d} \mathbf{v}^{(k)}(i_k) \text{ for all } 1 \le i_k \le N_k. \tag{14}$$

Here $\mathbf{v}^{(k)}(i_k)$ denotes the i_k-th element of vector $\mathbf{v}^{(k)}$.

Definition 5 (Tensor Rank). *The rank of $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{N_1 \times \cdots \times N_d}$ is the smallest positive integer $\bar{r}$, such that*

$$\boldsymbol{\mathcal{A}} = \sum_{j=1}^{\bar{r}} \mathbf{v}_j^{(1)} \circ \mathbf{v}_j^{(2)} \cdots \circ \mathbf{v}_j^{(d)}, \text{ with } \mathbf{v}_j^{(k)} \in \mathbb{R}^{N_k}. \tag{15}$$

It is attractive to perform tensor decomposition: given a small integer $r < \bar{r}$, approximate $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{N_1 \times \cdots \times N_d}$ by a rank-r tensor. Popular tensor decomposition algorithms include canonical decomposition [44]–[46] and Tuker decomposition [47], [48]. Canonical tensor decomposition aims to approximate $\boldsymbol{\mathcal{A}}$ by the sum of r rank-1 tensors [in the form of (15)] while minimizing the approximation error, which is normally implemented with alternating least square [45]. This decomposition scales well with the dimensionality d, but it is ill-posed for $d \geq 3$ [49]. Tucker decomposition aims to represent a tensor by a small core tensor and some matrix factors [47], [48]. This decomposition is based on singular value decomposition. It is robust, but the number of elements in the core tensor still grows exponentially with d.

Alternatively, tensor-train decomposition [39]–[41] approximates $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{N_1 \times \cdots \times N_d}$ by a low-rank tensor $\hat{\boldsymbol{\mathcal{A}}}$ with

$$\hat{\boldsymbol{\mathcal{A}}}(i_1, \cdots i_d) = \boldsymbol{\mathcal{G}}_1(:, i_1, :)\, \boldsymbol{\mathcal{G}}_2(:, i_1, :) \cdots \boldsymbol{\mathcal{G}}_d(:, i_d, :). \quad (16)$$

Here $\boldsymbol{\mathcal{G}}_k \in \mathbb{R}^{r_{k-1} \times N_k \times r_k}$, and $r_0 = r_d = 1$. By fixing the second index i_k, $\boldsymbol{\mathcal{G}}_k(:, i_k, :) \in \mathbb{R}^{r_{k-1} \times r_k}$ becomes a matrix (or vector when k equals 1 or d). To some extent, tensor-train decomposition have the advantages of both canonical tensor decomposition and Tuker decomposition: it is robust since each core tensor is obtained by a well-posed low-rank matrix decomposition [39]–[41]; it scales linearly with d since storing all core tensors requires only $O(Nr^2d)$ memory if we assume $N_k = N$ and $r_k = r$ for $k = 1, \cdots, d-1$. Given an error bound ϵ, the tensor train decomposition in (16) ensures

$$\left\| \boldsymbol{\mathcal{A}} - \hat{\boldsymbol{\mathcal{A}}} \right\|_F \leq \varepsilon \left\| \boldsymbol{\mathcal{A}} \right\|_F \quad (17)$$

while keeping r_k's as small as possible [39].

Definition 6 (TT-Rank). *In tensor-train decomposition (16) $\boldsymbol{\mathcal{G}}_k \in \mathbb{R}^{r_{k-1} \times N_k \times r_k}$ for $k = 1, \cdots d$. The vector $\vec{r} = [r_0, r_1, \cdots, r_d]$ is called TT-rank.*

Recently, tensor decomposition has shown promising applications in high-dimensional data and image compression [50]–[53], and in machine learning [54], [55]. In the uncertainty quantification community, some efficient high-dimensional stochastic PDE solvers have been developed based on canonical tensor decomposition [56]–[58] (which is called "Proper Generalized Decomposition" in some papers) and tensor-train decomposition [59]–[62]. In [63], a spectral tensor-train decomposition is proposed for high-dimensional function approximation.

III. ANOVA-Based Surrogate Model Extraction

In order to accelerate the low-level simulation, this section develops a sparse stochastic circuit/MEMS simulator based on anchored ANOVA (analysis of variance). Without of loss of generality, let $y = g(\vec{\xi})$ denote the output of a subsystem. We assume that y is a smooth function of the random parameters $\vec{\xi} \in \Omega \subseteq \mathbb{R}^d$ that describe the process variations.

A. ANOVA and Anchored ANOVA Decomposition

1) ANOVA: With ANOVA decomposition [34], [64], y can be written as

$$y = g(\vec{\xi}) = \sum_{s \subseteq \mathcal{I}} g_s(\vec{\xi}_s), \quad (18)$$

where s is a subset of the full index set $\mathcal{I} = \{1, 2, \cdots, d\}$. Let $\bar{s}$ be the complementary set of s such that $s \cup \bar{s} = \mathcal{I}$ and $s \cap \bar{s} = \emptyset$, and let $|s|$ be the number of elements in s. When $s = \{i_1, \cdots, i_{|s|}\} \neq \emptyset$, we set $\Omega_s = \Omega_{i_1} \otimes \cdots \otimes \Omega_{i_{|s|}}$, $\vec{\xi}_s = [\xi_{i_1}, \cdots, \xi_{i_{|s|}}] \in \Omega_s$ and have the Lebesgue measure

$$d\mu(\vec{\xi}_{\bar{s}}) = \prod_{k \in \bar{s}} (\rho_k(\xi_k)\, d\xi_k). \quad (19)$$

Then, $g_s(\vec{\xi}_s)$ in ANOVA decomposition (18) is defined recursively by the following formula

$$g_s(\vec{\xi}_s) = \begin{cases} \mathbb{E}\left(g(\vec{\xi})\right) = \int_{\Omega} g(\vec{\xi}) d\mu(\vec{\xi}) = g_0, \text{ if } s = \emptyset \\ \hat{g}_s(\vec{\xi}_s) - \sum_{t \subset s} g_t(\vec{\xi}_t)\ , \text{ if } s \neq \emptyset. \end{cases} \quad (20)$$

Here $\mathbb{E}$ is the expectation operator, $\hat{g}_s(\vec{\xi}_s) = \int_{\Omega_{\bar{s}}} g(\vec{\xi}) d\mu(\vec{\xi}_{\bar{s}})$, and the integration is computed for all elements except those in $\vec{\xi}_s$. From (20), we have the following intuitive results:

- g_0 is a constant term;
- if s=$\{j\}$, then $\hat{g}_s(\vec{\xi}_s) = \hat{g}_{\{j\}}(\xi_j)$, $g_s(\vec{\xi}_s) = g_{\{j\}}(\xi_j) = \hat{g}_{\{j\}}(\xi_j) - g_0$;
- if s=$\{j, k\}$ and $j < k$, then $\hat{g}_s(\vec{\xi}_s) = \hat{g}_{\{j,k\}}(\xi_j, \xi_k)$ and $g_s(\vec{\xi}_s) = \hat{g}_{\{j,k\}}(\xi_j, \xi_k) - g_{\{j\}}(\xi_j) - g_{\{k\}}(\xi_k) - g_0$;
- both $\hat{g}_s(\vec{\xi}_s)$ and $g_s(\vec{\xi}_s)$ are $|s|$-variable functions, and the decomposition (18) has 2^d terms in total.

Example 1. *Consider $y = g(\vec{\xi}) = g(\xi_1, \xi_2)$. Since $\mathcal{I} = \{1, 2\}$, its subset includes $\emptyset$, $\{1\}$, $\{2\}$ and $\{1, 2\}$. As a result, there exist four terms in the ANOVA decomposition (18):*

- *for $s = \emptyset$, $g_\emptyset(\vec{\xi}_\emptyset) = \mathbb{E}\left(g(\vec{\xi})\right) = g_0$ is a constant;*
- *for $s = \{1\}$, $g_{\{1\}}(\xi_1) = \hat{g}_{\{1\}}(\xi_1) - g_0$, and $\hat{g}_{\{1\}}(\xi_1) = \int_{\Omega_2} g(\vec{\xi}) \rho_2(\xi_2) d\xi_2$ is a univariate function of ξ_1;*
- *for $s = \{2\}$, $g_{\{2\}}(\xi_2) = \hat{g}_{\{2\}}(\xi_2) - g_0$, and $\hat{g}_{\{2\}}(\xi_2) = \int_{\Omega_1} g(\vec{\xi}) \rho_1(\xi_1) d\xi_1$ is a univariate function of ξ_2;*
- *for $s = \{1, 2\}$, $g_{\{1,2\}}(\xi_1, \xi_2) = \hat{g}_{\{1,2\}}(\xi_1, \xi_2) - g_{\{1\}}(\xi_1) - g_{\{2\}}(\xi_2) - g_0$. Since $\bar{s} = \emptyset$, we have $\hat{g}_{\{1,2\}}(\xi_1, \xi_2) = g(\vec{\xi})$, which is a bi-variate function.*

Since all terms in the ANOVA decomposition are mutually orthogonal [34], [64], we have

$$\mathbf{Var}\left(g(\vec{\xi})\right) = \sum_{s \subseteq \mathcal{I}} \mathbf{Var}\left(g_s(\vec{\xi}_s)\right) \quad (21)$$

where $\mathbf{Var}(\bullet)$ denotes the variance over the whole parameter space Ω. What makes ANOVA practically useful is that for many engineering problems, $g(\vec{\xi})$ is mainly influenced by the terms that depend only on a small number of variables, and thus it can be well approximated by a truncated ANOVA decomposition

$$g(\vec{\xi}) \approx \sum_{|s| \leq d_{\text{eff}}} g_s(\vec{\xi}_s),\ s \subseteq \mathcal{I} \quad (22)$$

where $d_{\text{eff}} \ll d$ is called the **effective dimension**.

Example 2. *Consider $y = g(\vec{\xi})$ with $d = 20$. In the full ANOVA decomposition (18), we need to compute over 10^6 terms, which is prohibitively expensive. However, if we set $d_{\text{eff}} = 2$, we have the following approximation*

$$g(\vec{\xi}) \approx g_0 + \sum_{j=1}^{20} g_j(\xi_j) + \sum_{1\le j<k\le 20} g_{j,k}(\xi_j, \xi_k) \tag{23}$$

which contains only 221 terms.

Unfortunately, it is still expensive to obtain the truncated ANOVA decomposition (22) due to two reasons. First, the high-dimensional integrals in (20) are expensive to compute. Second, the truncated ANOVA decomposition (22) still contains lots of terms when d is large. In the following, we introduce anchored ANOVA that solves the first problem. The second issue will be addressed in Section III-B.

2) Anchored ANOVA: In order to avoid the expensive multidimensional integral computation, [34] has proposed an efficient algorithm which is called anchored ANOVA in [33], [35], [36]. Assuming that ξ_k's have standard uniform distributions, anchored ANOVA first chooses a deterministic point called anchored point $\vec{q} = [q_1, \cdots, q_d] \in [0,1]^d$, and then replaces the Lebesgue measure with the Dirac measure

$$d\mu(\vec{\xi}_{\bar{s}}) = \prod_{k\in\bar{s}} \left(\delta\left(\xi_k - q_k\right) d\xi_k\right). \tag{24}$$

As a result, $g_0 = g(\vec{q})$, and

$$\hat{g}_s(\vec{\xi}_s) = g\left(\tilde{\xi}\right), \text{ with } \tilde{\xi}_k = \begin{cases} q_k, \text{ if } k \in \bar{s} \\ \xi_k, \text{ otherwise.} \end{cases} \tag{25}$$

Here $\tilde{\xi}_k$ denotes the k-th element of $\tilde{\xi} \in \mathbb{R}^d$, q_k is a fixed deterministic value, and ξ_k is a random variable. Anchored ANOVA was further extended to Gaussian random parameters in [35]. In [33], [36], [37], this algorithm was combined with stochastic collocation to efficiently solve high-dimensional stochastic partial differential equations.

Example 3. *Consider $y{=}g(\xi_1, \xi_2)$. With an anchored point $\vec{q} = [q_1, q_2]$, we have $g_0 = g(q_1, q_2)$, $\hat{g}_{\{1\}}(\xi_1) = g(\xi_1, q_2)$, $\hat{g}_{\{2\}}(\xi_2) = g(q_1, \xi_2)$ and $\hat{g}_{\{1,2\}}(\xi_1, \xi_2) = g(\xi_1, \xi_2)$. Computing these quantities does not involve any high-dimensional integrations.*

B. Adaptive Anchored ANOVA for Circuit/MEMS Problems

1) Extension to General Cases: In many circuit and MEMS problems, the process variations can be non-uniform and non-Gaussian. We show that anchored ANOVA can be applied to such general cases.

Observation: The anchored ANOVA in [34] can be applied if $\rho_k(\xi_k) > 0$ for any $\xi_k \in \Omega_k$.

Proof: Let u_k denote the cumulative density function for ξ_k, then u_k can be treated as a random variable uniformly distributed on $[0,1]$. Since $\rho_k(\xi_k) > 0$ for any $\xi_k \in \Omega_k$, there exists $\xi_k = \lambda_k(u_k)$ which maps u_k to ξ_k. Therefore, $g(\xi_1, \cdots, \xi_d) = g\left(\lambda_1(u_1), \cdots, \lambda_d(u_d)\right) = \psi(\vec{u})$ with $\vec{u} = [u_1, \cdots, u_d]$. Following (25), we have

$$\hat{\psi}_s(\vec{u}_s) = \psi\left(\tilde{u}\right), \text{ with } \tilde{u}_k = \begin{cases} p_k, \text{ if } k \in \bar{s} \\ u_k, \text{ otherwise,} \end{cases} \tag{26}$$

where $\vec{p} = [p_1, \cdots, p_d]$ is the anchor point for $\vec{u}$. The above result can be rewritten as

$$\hat{g}_s(\vec{\xi}_s) = g\left(\tilde{\xi}\right), \text{ with } \tilde{\xi}_k = \begin{cases} \lambda_k(q_k), \text{ if } k \in \bar{s} \\ \lambda_k(\xi_k), \text{ otherwise,} \end{cases} \tag{27}$$

from which we can obtain $g_s(\vec{\xi}_s)$ defined in (20). Consequently, the decomposition for $g(\vec{\xi})$ can be obtained by using $\vec{q} = [\lambda_1(p_1), \cdots, \lambda_d(p_d)]$ as an anchor point of $\vec{\xi}$. ■

Anchor point selection. It is is important to select a proper anchor point [36]. In circuit and MEMS applications, we find that $\vec{q} = \mathbb{E}(\vec{\xi})$ is a good choice.

2) Adaptive Implementation: In order to further reduce the computational cost, the truncated ANOVA decomposition (22) can be implemented in an adaptive way. Specifically, in practical computation we can ignore those terms that have small variance values. Such a treatment can produce a highly sparse generalized polynomial-chaos expansion.

For a given effective dimension $d_{\text{eff}} \ll d$, let

$$\mathcal{S}_k = \{s | s \subset \mathcal{I}, |s| = k\}, \; k = 1, \cdots d_{\text{eff}} \tag{28}$$

contain the initialized index sets for all k-variate terms in the ANOVA decomposition. Given an anchor point $\vec{q}$ and a threshold σ, starting from $k{=}1$, the main procedures of our ANOVA-based stochastic simulator are summarized below:

1) Compute g_0, which is a deterministic evaluation;
2) For every $s \in \mathcal{S}_k$, compute the low-dimensional function $g_s(\vec{\xi}_s)$ by stochastic testing. The importance of $g_s(\vec{\xi}_s)$ is measured as
$$\theta_s = \frac{\mathbf{Var}\left(g_s\left(\vec{\xi}_s\right)\right)}{\sum_{j=1}^{k} \sum_{\tilde{s}\in S_j} \mathbf{Var}\left(g_{\tilde{s}}\left(\vec{\xi}_{\tilde{s}}\right)\right)}. \tag{29}$$
3) Update the index sets if $\theta_s < \sigma$ for $s \in \mathcal{S}_k$. Specifically, for $k < j \le d_{\text{eff}}$, we check its index set $s' \in \mathcal{S}_j$. If s' contains all elements of s, then we remove s' from $\mathcal{S}_j$. Once s' is removed, we do not need to evaluate $g_{s'}(\vec{\xi}_{s'})$ in the subsequent computation.
4) Set $k{=}\, k+1$, and repeat steps 2) and 3) until $k = d_{\text{def}}$.

Example 4. *Let $y{=}g(\vec{\xi})$, $\vec{\xi} \in \mathbb{R}^{20}$ and $d_{\text{eff}} = 2$. Anchored ANOVA starts with*

$$\mathcal{S}_1 = \{\{j\}\}_{j=1,\cdots,20} \text{ and } \mathcal{S}_2 = \{\{j,k\}\}_{1\le j<k\le 20}.$$

For $k{=}1$, we first utilize stochastic testing to calculate $g_s(\vec{\xi}_s)$ and θ_s for every $s \in \mathcal{S}_1$. Assume

$$\theta_{\{1\}} > \sigma, \; \theta_{\{2\}} > \sigma, \text{ and } \theta_{\{j\}} < \sigma \text{ for all } j > 2,$$

implying that only the first two parameters are important to the output. Then, we only consider the coupling of ξ_1 and ξ_2 in $\mathcal{S}_2$, leading to

$$\mathcal{S}_2 = \{\{1,2\}\}.$$

Algorithm 1 Stochastic Testing Circuit/MEMS Simulator Based on Adaptive Anchored ANOVA.

1: Initialize $\mathcal{S}_k$'s and set $\beta = 0$;
2: At the anchor point, run a deterministic circuit/MEMS simulation to obtain g_0, and set $y = g_0$;
3: **for** $k = 1, \cdots, d_{\text{eff}}$ **do**
4: **for** each $s \in \mathcal{S}_k$ **do**
5: run stochastic testing simulator to get the generalized polynomial-chaos expansion of $\hat{g}_s(\vec{\xi}_s)$;
6: get the generalized polynomial-chaos expansion of $g_s(\vec{\xi}_s)$ according to (20);
7: update $\beta = \beta + \mathbf{Var}\left(g_s(\vec{\xi}_s)\right)$;
8: update $y = y + g_s(\vec{\xi}_s)$;
9: **end for**
10: **for** each $s \in \mathcal{S}_k$ **do**
11: $\theta_s = \mathbf{Var}\left(g_s(\vec{\xi}_s)\right)/\beta$;
12: **if** $\theta_s < \sigma$
13: for any index set $s' \in \mathcal{S}_j$ with $j > k$, remove s' from $\mathcal{S}_j$ if $s \subset s'$.
14: **end if**
15: **end for**
16: **end for**

Consequently, for $k = 2$ we only need to calculate one bivariate function $g_{\{1,2\}}(\xi_1, \xi_2)$, yielding

$$g\left(\vec{\xi}\right) \approx g_0 + \sum_{s\in S_1} g_s\left(\vec{\xi}_s\right) + \sum_{s\in S_2} g_s\left(\vec{\xi}_s\right) = g_0 + \sum_{j=1}^{20} g_{\{j\}}\left(\xi_j\right) + g_{\{1,2\}}\left(\xi_1, \xi_2\right).$$

The pseudo codes of our implementation are summarized in Alg. 1. Lines 10 to 15 shows how to adaptively select the index sets. Let the final size of $\mathcal{S}_k$ be $|\mathcal{S}_k|$ and the total polynomial order in the stochastic testing simulator be p, then the total number of samples used in Alg. 1 is

$$N = 1 + \sum_{k=1}^{d_{\text{eff}}} |\mathcal{S}_k| \frac{(k+p)!}{k!p!}. \tag{30}$$

Note that all univariate terms in ANOVA (i.e., $|s| = 1$) are kept in our implementation. For most circuit and MEMS problems, setting the effective dimension as 2 or 3 can achieve a high accuracy due to the weak couplings among different random parameters. For many cases, the univariate terms dominate the output of interest, leading to a near-linear complexity with respect to the parameter dimensionality d.

Remarks. Anchored ANOVA works very well for a large class of MEMS and circuit problems. However, in practice we also find a small number of examples (e.g., CMOS ring oscillators) that cannot be solved efficiently by the proposed algorithm, since many random variables affect significantly the output of interest. For such problems, it is possible to reduce the number of dominant random variables by a linear transform [65] before applying anchored ANOVA. Other techniques such as compressed sensing can also be utilized to extract highly sparse surrogate models [66]–[69] in the low-level simulation of our proposed hierarchical framework.

3) Global Sensitivity Analysis: Since each term $g_s(s_s)$ is computed by stochastic testing, Algorithm 1 provides a sparse generalized polynomial-chaos expansion for the output of interest: $y = \sum_{|\vec{\alpha}|\le p} y_{\vec{\alpha}} H_{\vec{\alpha}}(\vec{\xi})$, where most coefficients are zero. From this result, we can identify how much each parameter contributes to the output by global sensitivity analysis. Two kinds of sensitivity information can be used to measure the importance of parameter ξ_k: the main sensitivity S_k and total sensitivity T_k, as computed below:

$$S_k = \frac{\sum_{\alpha_k\neq 0, \alpha_{j\neq k}=0} |y_{\vec{\alpha}}|^2}{\mathbf{Var}(y)}, \quad T_k = \frac{\sum_{\alpha_k\neq 0} |y_{\vec{\alpha}}|^2}{\mathbf{Var}(y)}. \tag{31}$$

IV. Enabling High-Level Simulation by Tensor-Train Decomposition

In this section, we show how to accelerate the high-level non-Monte-Carlo simulation by handling the obtained high-dimensional surrogate models with tensor-train decomposition [39]–[41].

A. Tensor-Based Three-Term Recurrence Relation

In order to obtain the orthonormal polynomials and Gauss quadrature points/weights of ζ, we must implement the three-term recurrence relation in (7). The main bottleneck is to compute the integrals in (8), since the probability density function of ζ is unknown.

For simplicity, we rewrite the integrals in (8) as $\mathbb{E}(q(\zeta))$, with $q(\zeta) = \phi_j^2(\zeta)$ or $q(\zeta) = \zeta\phi_j^2(\zeta)$. Since the probability density function of ζ is not given, we compute the integral in the parameter space Ω:

$$\mathbb{E}\left(q\left(\zeta\right)\right) = \int_{\Omega} q\left(f\left(\vec{\xi}\right)\right) \rho(\vec{\xi}) d\xi_1 \cdots d\xi_d, \tag{32}$$

where $f(\vec{\xi})$ is a sparse generalized polynomial-chaos expansion for ζ obtained by

$$\zeta = f(\vec{\xi}) = \frac{(y - \mathbb{E}(y))}{\sqrt{\mathbf{Var}(y)}} = \sum_{|\vec{\alpha}|\le p} \hat{y}_{\vec{\alpha}} H_{\vec{\alpha}}(\vec{\xi}). \tag{33}$$

We compute the integral in (32) with the following steps:

1) We utilize a multi-dimensional Gauss quadrature rule:

$$\mathbb{E}\left(q\left(\zeta\right)\right) \approx \sum_{i_1=1}^{m_1} \cdots \sum_{i_d=1}^{m_d} q\left(f\left(\xi_1^{i_1}, \cdots, \xi_d^{i_d}\right)\right) \prod_{k=1}^{d} w_k^{i_k} \tag{34}$$

where m_k is the number of quadrature points for ξ_k, $(\xi_k^{i_k}, w_k^{i_k})$ denotes the i_k-th Gauss quadrature point and weight.

2) We define two d-mode tensors $\boldsymbol{\mathcal{Q}}, \boldsymbol{\mathcal{W}} \in \mathbb{R}^{m_1\times m_2\cdots\times m_d}$, with each element defined as

$$\begin{aligned} \boldsymbol{\mathcal{Q}}\left(i_1, \cdots i_d\right) &= q\left(f\left(\xi_1^{i_1}, \cdots, \xi_d^{i_d}\right)\right), \\ \boldsymbol{\mathcal{W}}\left(i_1, \cdots i_d\right) &= \prod_{k=1}^{d} w_k^{i_k}, \end{aligned} \tag{35}$$

for $1 \le i_k \le m_k$. Now we can rewrite (34) as the inner product of $\boldsymbol{\mathcal{Q}}$ and $\boldsymbol{\mathcal{W}}$:

$$\mathbb{E}\left(q\left(\zeta\right)\right) \approx \left\langle \boldsymbol{\mathcal{Q}}, \boldsymbol{\mathcal{W}} \right\rangle. \tag{36}$$

For simplicity, we set $m_k{=}m$ in this manuscript.

The cost of computing the tensors and the tensor inner product is $O(m^d)$, which becomes intractable when d is large. Fortunately, both $\boldsymbol{\mathcal{Q}}$ and $\boldsymbol{\mathcal{W}}$ have low tensor ranks in our applications, and thus the high-dimensional integration (32) can be computed very efficiently in the following way:

1) **Low-rank representation of $\boldsymbol{\mathcal{W}}$.** $\boldsymbol{\mathcal{W}}$ can be written as a rank-1 tensor
$$\boldsymbol{\mathcal{W}} = \mathbf{w}^{(1)} \circ \mathbf{w}^{(2)} \cdots \circ \mathbf{w}^{(d)}, \tag{37}$$
where $\mathbf{w}^{(k)} = [w_k^1; \cdots; w_k^m] \in \mathbb{R}^{m\times 1}$ contains all Gauss quadrature weights for parameter ξ_k. Clearly, now we only need $O(md)$ memory to store $\boldsymbol{\mathcal{W}}$.
2) **Low-rank approximation for $\boldsymbol{\mathcal{Q}}$.** $\boldsymbol{\mathcal{Q}}$ can be well approximated by $\hat{\boldsymbol{\mathcal{Q}}}$ with high accuracy in a tensor-train format [39]–[41]:
$$\hat{\boldsymbol{\mathcal{Q}}}(i_1, \cdots i_d) = \boldsymbol{\mathcal{G}}_1(:, i_1, :)\, \boldsymbol{\mathcal{G}}_2(:, i_1, :) \cdots \boldsymbol{\mathcal{G}}_d(:, i_d, :) \tag{38}$$
with a pre-selected error bound ϵ such that
$$\left\| \boldsymbol{\mathcal{Q}} - \hat{\boldsymbol{\mathcal{Q}}} \right\|_F \le \varepsilon \left\| \boldsymbol{\mathcal{Q}} \right\|_F. \tag{39}$$
For many circuit and MEMS problems, a tensor train with very small TT-ranks can be obtained even when $\epsilon = 10^{-12}$ (which is very close to the machine precision).
3) **Fast computation of (36).** With the above low-rank tensor representations, the inner product in (36) can be accurately estimated as
$$\left\langle \hat{\boldsymbol{\mathcal{Q}}}, \boldsymbol{\mathcal{W}} \right\rangle = \mathbf{T}_1 \cdots \mathbf{T}_d, \text{ with } \mathbf{T}_k = \sum_{i_k=1}^{m} w_k^{i_k} \boldsymbol{\mathcal{G}}_k(:, i_k, :) \tag{40}$$
Now the cost of computing the involved high-dimensional integration dramatically reduces to $O(dmr^2)$, which only linearly depends the parameter dimensionality d.

B. Efficient Tensor-Train Computation

Now we discuss how to obtain a low-rank tensor train. An efficient implementation called **TT_cross** is described in [41] and included in the public-domain MATALB package **TT_Toolbox** [70]. In **TT_cross**, Skeleton decomposition is utilized to compress the TT-rank r_k by iteratively searching a rank-r_k maximum-volume submatrix when computing $\boldsymbol{\mathcal{G}}_k$. A major advantage of **TT_cross** is that we do not need to know $\boldsymbol{\mathcal{Q}}$ *a-priori*. Instead, we only need to specify how to evaluate the element $\boldsymbol{\mathcal{Q}}(i_1, \cdots, i_d)$ for a given index $(i_1, \cdots, i_d)$. As shown in [41], with Skeleton decompositions a tensor-train decomposition needs $O(ldmr^2)$ element evaluations, where l is the number of iterations in a Skeleton decomposition. For example, when $l = 10$, $d = 50$, $m = 10$ and $r = 4$ we may need up to 10^5 element evaluations, which can take about one hour since each element of $\boldsymbol{\mathcal{Q}}$ is a high-order polynomial function of many bottom-level random variables $\vec{\xi}$.

In order to make the tensor-train decomposition of $\boldsymbol{\mathcal{Q}}$ fast, we employ some tricks to evaluate more efficiently each element of $\boldsymbol{\mathcal{Q}}$. The details are given below.

- **Fast evaluation of $\boldsymbol{\mathcal{Q}}(i_1, \cdots, i_d)$.** In order to reduce the cost of evaluating $\boldsymbol{\mathcal{Q}}(i_1, \cdots, i_d)$, we first construct a low-rank tensor train $\hat{\boldsymbol{\mathcal{A}}}$ for the intermediate-level random parameter ζ, such that
$$\left\| \boldsymbol{\mathcal{A}} - \hat{\boldsymbol{\mathcal{A}}} \right\|_F \le \varepsilon \left\| \boldsymbol{\mathcal{A}} \right\|_F, \ \boldsymbol{\mathcal{A}}(i_1, \cdots, i_d) = f\left(\xi_1^{i_1}, \cdots, \xi_d^{i_d}\right).$$
Once $\hat{\boldsymbol{\mathcal{A}}}$ is obtained, $\boldsymbol{\mathcal{Q}}(i_1, \cdots, i_d)$ can be evaluated by
$$\boldsymbol{\mathcal{Q}}(i_1, \cdots, i_d) \approx q\left(\hat{\boldsymbol{\mathcal{A}}}(i_1, \cdots, i_d)\right), \tag{41}$$
which reduces to a cheap low-order univariate polynomial evaluation. However, computing $\hat{\boldsymbol{\mathcal{A}}}(i_1, \cdots, i_d)$ by directly evaluating $\boldsymbol{\mathcal{A}}(i_1, \cdots, i_d)$ in **TT_cross** can be time-consuming, since $\zeta = f(\vec{\xi})$ involves many multivariate basis functions.
- **Fast evaluation of $\boldsymbol{\mathcal{A}}(i_1, \cdots, i_d)$.** The evaluation of $\boldsymbol{\mathcal{A}}(i_1, \cdots, i_d)$ can also be accelerated by exploiting the special structure of $f(\vec{\xi})$. It is known that the generalized polynomial-chaos basis of $\vec{\xi}$ is
$$H_{\vec{\alpha}}\left(\vec{\xi}\right) = \prod_{k=1}^{d} \varphi_{\alpha_k}^{(k)}(\xi_k), \ \vec{\alpha} = [\alpha_1, \cdots, \alpha_d] \tag{42}$$
where $\varphi_{\alpha_k}^{(k)}(\xi_k)$ is the degree-α_k orthonormal polynomial of ξ_k, with $0 \le \alpha_k \le p$. We first construct a 3-mode tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{d\times(p+1)\times m}$ indexed by $(k, \alpha_k + 1, i_k)$ with
$$\boldsymbol{\mathcal{X}}(k, \alpha_k + 1, i_k) = \varphi_{\alpha_k}^{(k)}\left(\xi_k^{i_k}\right) \tag{43}$$
where $\xi_k^{i_k}$ is the i_k-th Gauss quadrature point for parameter ξ_k [as also used in (34)]. Then, each element of $\boldsymbol{\mathcal{A}}(i_1, \cdots, i_d)$ can be calculated efficiently as
$$\boldsymbol{\mathcal{A}}(i_1, \cdots, i_d) = \sum_{|\vec{\alpha}|<p} \vec{y}_{\vec{\alpha}} \prod_{k=1}^{d} \boldsymbol{\mathcal{X}}(k, \alpha_k + 1, i_k) \tag{44}$$
without evaluating the multivariate polynomials. Constructing $\boldsymbol{\mathcal{X}}$ does not necessarily need $d(p+1)m$ polynomial evaluations, since the matrix $\boldsymbol{\mathcal{X}}(k, :, :)$ can be reused for any other parameter ξ_j that has the same type of distribution with ξ_k.

In summary, we compute a tensor-train decomposition for $\boldsymbol{\mathcal{Q}}$ as follows: 1) we construct the 3-mode tensor $\boldsymbol{\mathcal{X}}$ defined in (43); 2) we call **TT_cross** to compute $\hat{\boldsymbol{\mathcal{A}}}$ as a tensor-train decomposition of $\boldsymbol{\mathcal{A}}$, where (44) is used for fast element evaluation; 3) we call **TT_cross** again to compute $\hat{\boldsymbol{\mathcal{Q}}}$, where (41) is used for the fast element evaluation of $\boldsymbol{\mathcal{Q}}$. With the above fast tensor element evaluations, the computation time of **TT_cross** can be reduced from dozens of minutes to several seconds to generate some accurate low-rank tensor trains for our high-dimensional surrogate models.

C. Algorithm Summary

Given the Gauss quadrature rule for each bottom-level random parameter ξ_k, our tensor-based three-term recurrence relation for an intermediate-level random parameter ζ is summarized in Alg. 2. This procedure can be repeated for all ζ_i's to obtain their univariate generalized polynomial-chaos basis

Algorithm 2 Tensor-based generalized polynomial-chaos basis and Gauss quadrature rule construction for ζ.

1: Initialize: $\phi_0(\zeta) = \pi_0(\zeta) = 1$, $\phi_1(\zeta) = \pi_1(\zeta) = \zeta$, $\kappa_0 = \kappa_1 = 1$, $\gamma_0 = 0$, $a = 1$;
2: Compute a low-rank tensor train $\hat{\boldsymbol{\mathcal{A}}}$ for ζ;
3: Compute a low-rank tensor train $\hat{\boldsymbol{\mathcal{Q}}}$ for $q(\zeta) = \zeta^3$, and obtain $\gamma_1 = \left\langle \hat{\boldsymbol{\mathcal{Q}}}, \boldsymbol{\mathcal{W}} \right\rangle$ via (40);
4: **for** $j = 2, \cdots, p$ **do**
5: get $\pi_j(\zeta) = (\zeta - \gamma_{j-1})\pi_{j-1}(\zeta) - \kappa_{j-1}\pi_{j-2}(\zeta)$;
6: construct a low-rank tensor train $\hat{\boldsymbol{\mathcal{Q}}}$ for $q(\zeta) = \pi_j^2(\zeta)$, and compute $\hat{a} = \left\langle \hat{\boldsymbol{\mathcal{Q}}}, \boldsymbol{\mathcal{W}} \right\rangle$ via (40) ;
7: $\kappa_j = \hat{a}/a$, and update $a = \hat{a}$;
8: construct a low-rank tensor train $\hat{\boldsymbol{\mathcal{Q}}}$ for $q(\zeta) = \zeta\pi_j^2(\zeta)$, and compute $\gamma_j = \left\langle \hat{\boldsymbol{\mathcal{Q}}}, \boldsymbol{\mathcal{W}} \right\rangle / a$;
9: normalization: $\phi_j(\zeta) = \frac{\pi_j(\zeta)}{\sqrt{\kappa_0 \cdots \kappa_j}}$;
10: **end for**
11: Form matrix $\mathbf{J}$ in (10);
12: Eigenvalue decomposition: $\mathbf{J} = \mathbf{U}\Sigma\mathbf{U}^T$;
13: Compute the Gauss-quadrature abscissa $\zeta^j = \Sigma(j, j)$ and weight $w^j = (\mathbf{U}(1, j))^2$ for $j = 1, \cdots, p+1$;

functions and Gauss quadrature rules, and then the stochastic testing simulator [14]–[16] (and any other standard stochastic spectral method [7]–[9]) can be employed to perform high-level stochastic simulation.

Remarks. 1) If the outputs of a group of subsystems are identically independent, we only need to run Alg. 2 once and reuse the results for the other subsystems in the group. 2) When there exist many subsystems, our ANOVA-based stochastic solver may also be utilized to accelerate the high-level simulation.

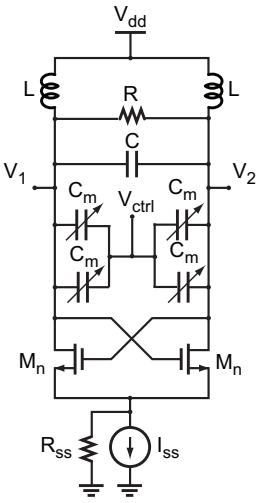
V. Numerical Results

In this section, we verify the proposed algorithm on a MEMS/IC co-design example with high-dimensional random parameters. All simulations are run in MATLAB and executed on a 2.4GHz laptop with 4GB memory.

A. MEMS/IC Example

In order to demonstrate the application of our hierarchical uncertainty quantification in high-dimensional problems, we consider the oscillator circuit shown in Fig. 3. This oscillator has four identical RF MEMS switches acting as tunable capacitors. The MEMS device used in this paper is a prototyping model of the RF MEMS capacitor reported in [71], [72].

Since the MEMS switch has a symmetric structure, we construct a model for only half of the design, as shown in Fig. 4. The simulation and measurement results in [42] show that the pull-in voltage of this MEMS switch is about 37 V. When the control voltage is far below the pull-in voltage, the MEMS capacitance is small and almost constant. In this paper, we set the control voltage to 2.5 V, and thus the MEMS switch can be regarded as a small linear capacitor. As already



Parameter	Value
V_{dd}	2.5V
V_{ctrl}	0-2.5V
C	10.83fF
R	110Ω
L	8.8nH
$(W/L)_n$	4/0.25
I_{ss}	0.5mA
R_{ss}	$10^6\Omega$

Fig. 3. Schematic of the oscillator circuit with 4 MEMS capacitors (denoted as $C_{\rm m}$), with 184 random parameters in total.

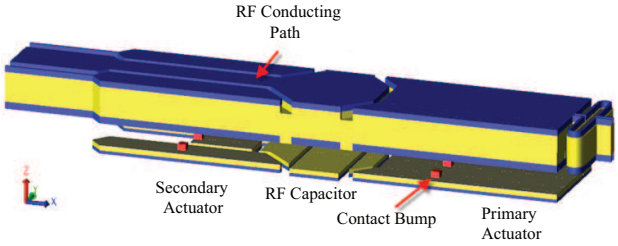


Fig. 4. 3-D schematic of the RF MEMS capacitor.

shown in [73], the performance of this MEMS switch can be influenced significantly by process variations.

In our numerical experiments, we use 46 independent random parameters with Gaussian and Gamma distributions to describe the material (e.g, conductivity and dielectric constants), geometric (e.g., thickness of each layer, width and length of each mechanical component) and environmental (e.g., temperature) uncertainties of each switch. For each random parameter, we assume that its standard deviation is 3% of its mean value. In the whole circuit, we have 184 random parameters in total. Due to such high dimensionality, simulating this circuit by stochastic spectral methods is a challenging task.

In the following experiments, we simulate this challenging design case using our proposed hierarchical stochastic spectral methods. We also compare our algorithm with other two kinds of hierarchical approaches listed in Table I. In Method 1, both low-level and high-level simulations use Monte Carlo, as suggested by [32]. In Method 2, the low-level simulation uses our ANOVA-based sparse simulator (Alg. 1), and the high-level simulation uses Monte Carlo.

B. Surrogate Model Extraction

In order to extract an accurate surrogate model for the MEMS capacitor, Alg. 1 is implemented in the commercial

TABLE II
SURROGATE MODEL EXTRACTION WITH DIFFERENT σ VALUES.

σ	# $\|s\|=1$	# $\|s\|=2$	# $\|s\|=3$	# ANOVA terms	# nonzero gPC terms	# samples
0.5	46	0	0	47	81	185
0.1 to 10^{-3}	46	3	0	50	90	215
10^{-4}	46	10	1	58	112	305
10^{-5}	46	21	1	69	144	415

TABLE I
DIFFERENT HIERARCHICAL SIMULATION METHODS.

Method	Low-level simulation	High-level simulation
Proposed	Alg. 1	stochastic testing [15]
Method 1 [32]	Monte Carlo	Monte Carlo
Method 2	Alg. 1	Monte Carlo

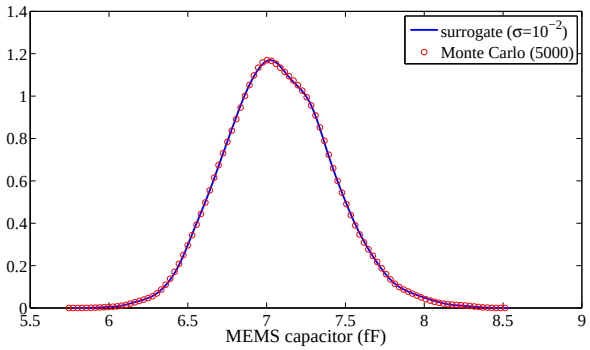


Fig. 5. Comparison of the density functions obtained by our surrogate model and by 5000-sample Monte Carlo analysis of the original MEMS equation.

network-based MEMS simulation tool MEMS+ [74] of Coventor Inc. Each MEMS switch is described by a stochastic differential equation [c.f. (1)] with consideration of process variations. In order to compute the MEMS capacitor, we can ignore the derivative terms and solve for the static solutions.

By setting $\sigma = 10^{-2}$, our ANOVA-based stochastic MEMS simulator generates a sparse 3rd-order generalized polynomial chaos expansion with only 90 non-zero coefficients, requiring only 215 simulation samples and 8.5 minutes of CPU time in total. This result has only 3 bivariate terms and no three-variable terms in ANOVA decomposition, due to the very weak couplings among different random parameters. Setting $\sigma = 10^{-2}$ can provide a highly accurate generalized polynomial chaos expansion for the MEMS capacitor, which has a relative error around 10^{-6} (in the L_2 sense) compared to that obtained by setting $\sigma = 10^{-5}$.

By evaluating the surrogate model and the original model (by simulating the original MEMS equation) with 5000 samples, we have obtained the same probability density curves shown in Fig. 5. Note that using the standard stochastic testing simulator [14]–[16] requires 18424 basis functions and simulation samples for this high-dimensional example, which is prohibitively expensive on a regular computer. When the effective dimension d_{eff} is set as 3, there should be 16262 terms in the truncated ANOVA decomposition (22). However, due to the weak couplings among different random parameters, only 90 of them are non-zero.

We can get surrogate models with different accuracies by changing the threshold σ. Table II has listed the number of obtained ANOVA terms, the number of non-zero generalized polynomial chaos (gPC) terms and the number of required simulation samples for different values of σ. From this table, we have the following observations:

1) When σ is large, only 46 univariate terms (i.e., the terms with $|s| = 1$) are obtained. This is because the variance of all univariate terms are regarded as small, and thus all multivariate terms are ignored.
2) When σ is reduced (for example, to 0.1), three dominant bivariate terms (with $|s| = 2$) are included by considering the coupling effects of the three most influential random parameters. Since the contributions of other parameters are insignificant, the result does not change even if σ is further decreased to 10^{-3}.
3) A three-variable term (with $|s| = 3$) and some bivariate coupling terms among other parameters can only be captured when σ is reduced to 10^{-4} or below. In this case, the effect of some non-dominant parameters can be captured.

Fig. 6 shows the global sensitivity of this MEMS capacitor with respect to all 46 random parameters. The output is

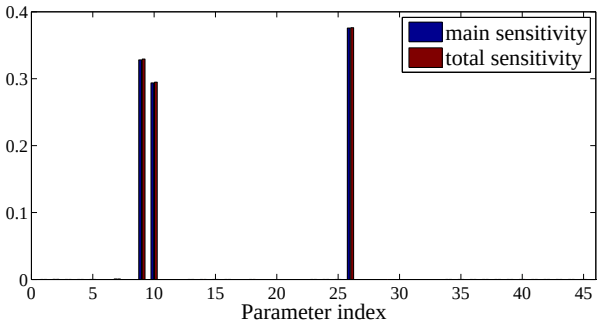


Fig. 6. Main and total sensitivities of different random parameters for the RF MEMS capacitor.

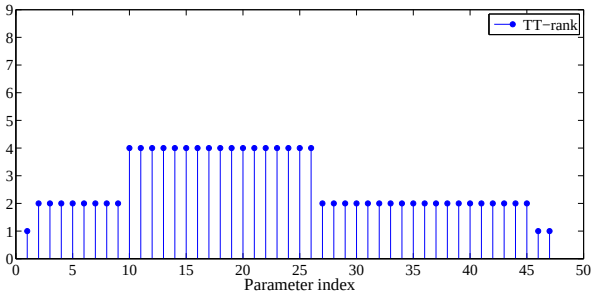


Fig. 7. TT-rank for the surrogate model of the RF MEMS capacitor.

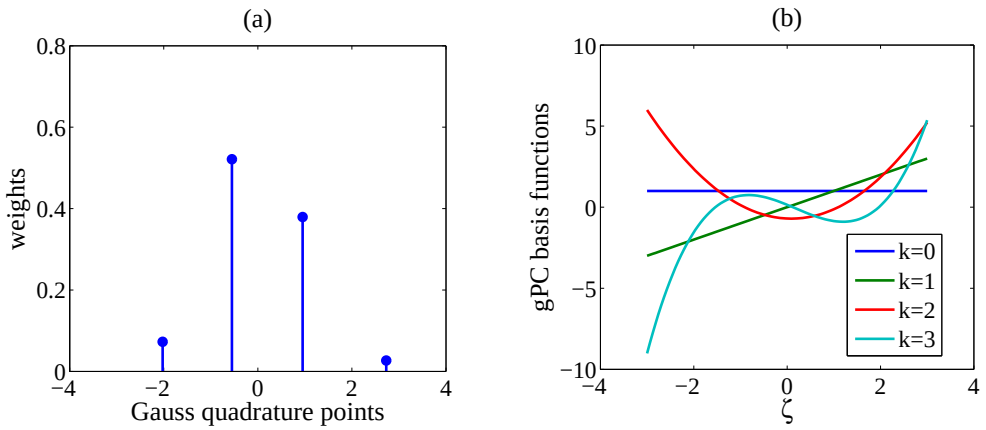


Fig. 8. (a) Gauss quadrature rule and (b) generalized polynomial chaos (gPC) basis functions for the RF MEMS capacitor.

dominated by only 3 parameters. The other 43 parameters contribute to only 2% of the capacitor's variance, and thus their main and total sensitivities are almost invisible in Fig. 6. This explains why the generalized polynomial-chaos expansion is highly sparse. Similar results have already been observed in the statistical analysis of CMOS analog circuits [1].

C. High-Level Simulation

The surrogate model obtained with $\sigma = 10^{-2}$ is imported into the stochastic testing circuit simulator described in [14]–[16] for high-level simulation. At the high-level, we have the following differential equation to describe the oscillator:

$$\frac{d\vec{q}\left(\vec{x}(t,\vec{\zeta}),\vec{\xi}\right)}{dt} + \vec{f}\left(\vec{x}(t,\vec{\zeta}),\vec{\zeta},u\right) = 0 \tag{45}$$

where the input signal u is constant, $\vec{\zeta}$=$[\zeta_1,\cdots,\zeta_4] \in \mathbb{R}^4$ are the intermediate-level random parameters describing the four MEMS capacitors. Since the oscillation period $T(\vec{\zeta})$ now depends on the MEMS capacitors, the periodic steady-state can be written as $\vec{x}(t,\zeta) = \vec{x}(t+T(\vec{\zeta}),\zeta)$. We simulate the stochastic oscillator by the following steps [15]:

1) Choose a constant $T_0 > 0$ to define an unknown scaling factor $a(\vec{\zeta}) = T(\vec{\zeta})/T_0$ and a scaled time axis $\tau = t/\alpha(\vec{\zeta})$. With this scaling factor, we obtain a reshaped waveform $\vec{z}(\tau,\vec{\zeta}) = \vec{x}(t/a(\vec{\zeta}),\vec{\zeta})$. At the steady state, we have $\vec{z}(\tau,\vec{\zeta}) = \vec{z}(\tau+T_0,\vec{\zeta})$. In other words, the reshaped waveform has a period T_0 independent of $\vec{\zeta}$.
2) Rewrite (45) on the scaled time axis:

$$\frac{d\vec{q}\left(\vec{z}(\tau,\vec{\zeta}),\vec{\xi}\right)}{d\tau} + a(\vec{\zeta})\vec{f}\left(\vec{z}(\tau,\vec{\zeta}),\vec{\zeta},u\right) = 0. \tag{46}$$

3) Approximate $\vec{z}(\tau,\vec{\zeta})$ and $a(\vec{\zeta})$ by generalized polynomial chaos expansions of $\vec{\zeta}$. Then, convert (46) to a larger-scale deterministic equation by stochastic testing. Solve the resulting deterministic equation by shooting Newton with a phase constraints, which would provide the coefficients in the generalized polynomial-chaos expansions of $\vec{z}(\tau,\vec{\zeta})$ and $\alpha(\vec{\zeta})$ [15].
4) Map $\vec{z}(\tau,\vec{\zeta})$ to the original time axis, we obtain the periodic steady state of $\vec{x}(t,\zeta)$.

In order to apply stochastic testing in Step 3), we need to compute some specialized orthonormal polynomials and Gauss quadrature points for each intermediate-level parameter ζ_i. We use 9 quadrature points for each bottom-level parameter ξ_k to evaluate the high-dimensional integrals involved in the three-term recurrence relation. This leads to 9^{46} function evaluations at all quadrature points, which is prohibitively expensive.

To handle the high-dimensional MEMS surrogate models, the following tensor-based procedures are employed:

- With Alg. 2, a low-rank tensor train of ζ_1 is first constructed for an MEMS capacitor. For most dimensions the rank is only 2, and the highest rank is 4, as shown in Fig. 7.
- Using the obtained tensor train, the Gauss quadrature points and generalized polynomial chaos basis functions are efficiently computed, as plotted in Fig. 8.

The total CPU time for constructing the tensor trains and computing the basis functions and Gauss quadrature points/weights is about 40 seconds in MATALB. If we directly evaluate the high-dimensional multivariate generalized polynomial-chaos expansion, the three-term recurrence relation requires almost 1 hour. The obtained results can be reused for all MEMS capacitors since they are independently identical.

With the obtained basis functions and Gauss quadrature points/weights for each MEMS capacitor, the stochastic periodic steady-state solver [15] is called at the high level to simulate the oscillator. Since there are 4 intermediate-level parameters ζ_i's, only 35 basis functions and testing samples are required for a 3rd-order generalized polynomial-chaos expansion, leading to a simulation cost of only 56 seconds in MATLAB.

Fig. 9 shows the waveforms from our algorithm at the scaled time axis $\tau = t/a(\vec{\zeta})$. The high-level simulation generates a generalized polynomial-chaos expansion for all nodal voltages, branch currents and the exact parameter-dependent period. Evaluating the resulting generalized polynomial-chaos expansion with 5000 samples, we have obtained the density function of the frequency, which is consistent with those from

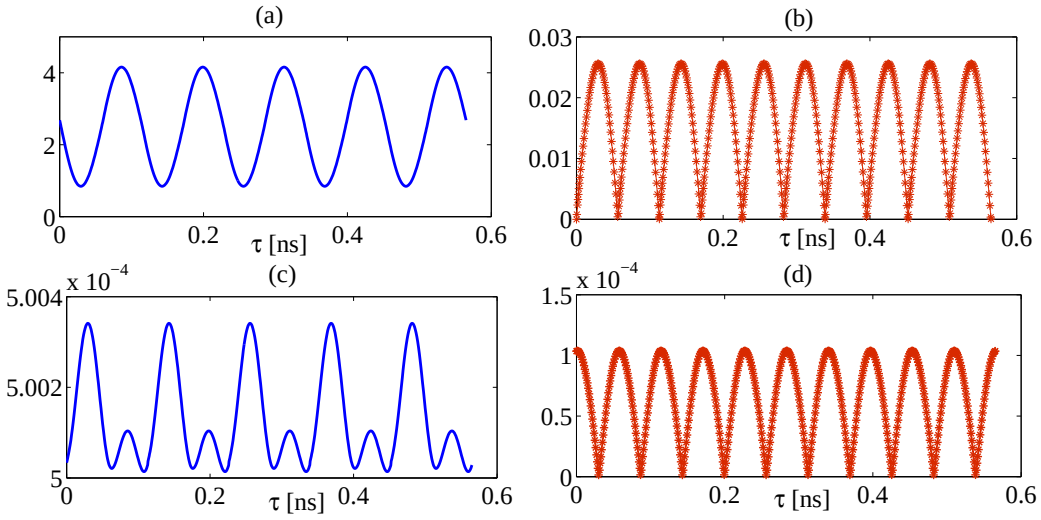


Fig. 9. Simulated waveforms on the scaled time axis $\tau = t/a(\vec{\zeta})$. (a) and (b): the mean and standard deviation of V_{out1} (unit: V), respectively; (c) and (d): the mean and standard deviation of the current (unit: A) from V_{dd}, respectively.

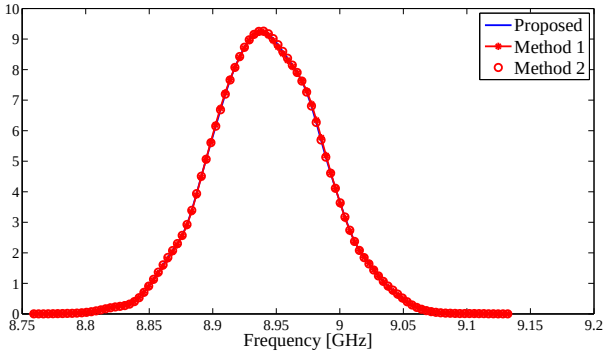


Fig. 10. Probability density functions of the oscillation frequency.

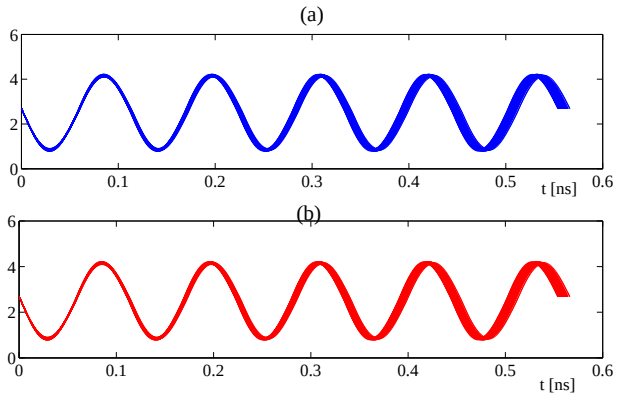


Fig. 11. Realization of the output voltages (unit: volt) at 100 bottom-level samples, generated by (a) proposed method and (b) Method 1.

Method 1 (using 5000 Monte Carlo samples at both levels) and Method 2 (using Alg. 1 at the low level and using 5000 Monte-Carlo samples at the high level), as shown in Fig. 10.

In order to show the variations of the waveform, we further plot the output voltages for 100 bottom-level random samples. As shown in Fig. 11, the results from our proposed method and from Method 1 are indistinguishable from each other.

D. Complexity Analysis

Table III has summarized the performances of all three methods. In all Monte Carlo analysis, 5000 random samples are utilized. If Method 1 [32] is used, Monte Carlo has to be repeatedly used for each MEMS capacitor, leading to extremely long CPU time due to the slow convergence. If Method 2 is used, the efficiency of the low-level surrogate model extraction can be improved due to the employment of generalized polynomial-chaos expansion, but the high-level simulation is still time-consuming. Since our proposed technique utilizes fast stochastic testing algorithms at both levels, this high-dimensional example can be simulated at very low computational cost, leading to 92× speedup over Method 1 and 14× speedup over Method 2.

VI. Conclusions and Future Work

This paper has proposed a framework to accelerate the hierarchical uncertainty quantification of stochastic circuits/systems with high-dimensional subsystems. We have developed an ANOVA-based stochastic testing simulator to accelerate the low-level simulation, and a tensor-based technique for handling high-dimensional surrogate models at the high level. Both algorithms have a linear (or near-linear) complexity with respect to the parameter dimensionality. Our simulator has been tested on an oscillator circuit with four MEMS capacitors and totally 184 random parameters, achieving highly accurate results at the cost of 10-min CPU time in MATLAB. In such example, our method is over 92× faster than the hierarchical Monte Carlo method developed in [32], and is about 14× faster than the method that uses ANOVA-based solver at the low level and Monte Carlo at the high level.

There are lots of problems worth investigation in the direction of hierarchical uncertainty quantification. Some unsolved important questions include:

1) How to extract a high-dimensional surrogate model such

TABLE III
CPU TIMES OF DIFFERENT HIERARCHICAL STOCHASTIC SIMULATION ALGORITHMS.

Simulation Method	Low level		High level		Total simulation cost
	Method	CPU time	Method	CPU time	
Proposed	Alg. 1	8.5 min	stochastic testing	1.5 minute	Low (10 min)
Method 1	Monte Carlo	13.2 h	Monte Carlo	2.2 h	High (15.4 h)
Method 2	Alg. 1	8.5 min	Monte Carlo	2.2 h	Medium (2.3 h)

that the tensor rank is as small as possible (or the tensor rank is below a provided upper bound)?

2) How to perform non-Monte-Carlo hierarchical uncertainty quantification when the outputs of different blocks are correlated?
3) How to perform non-Monte-Carlo hierarchical uncertainty quantification when y_i depends on some varying variables (e.g., time and frequency)?

ACKNOWLEDGMENTS

The authors would like to thank Coventor Inc. for providing the MEMS+ license and the MEMS switch design files. We would like to thank Shawn Cunningham and Dana Dereus of Wispry for providing access to the MEMS switch data. We are grateful to Dr. Giovanni Marucci for providing the oscillator design parameters, as well as Prof. Paolo Maffezzoni and Prof. Ibrahim Elfadel for their technical suggestions.

REFERENCES

[1] Z. Zhang, X. Yang, G. Marucci, P. Maffezzoni, I. M. Elfadel, G. Karniadakis, and L. Daniel, "Stochastic testing simulator for integrated circuits and MEMS: Hierarchical and sparse techniques," in *Proc. IEEE Custom Integrated Circuits Conf.* San Jose, CA, Sept. 2014.

[2] D. S. Boning, "Variation," *IEEE Trans. Semiconductor Manufacturing*, vol. 21, no. 1, pp. 63–71, Feb 2008.

[3] L. Yu, S. Saxena, C. Hess, A. Elfadel, D. A. Antoniadis, and D. S. Boning, "Remembrance of transistors past: Compact model parameter extraction using Bayesian inference and incomplete new measurements," in *Proc. Design Automation Conf.* San Francisco, CA, Jun 2014, pp. 1–6.

[4] L. Yu, S. Saxena, C. Hess, I. M. Elfadel, D. A. Antoniadis, and D. S. Boning, "Efficient performance estimation with very small sample size via physical subspace projection and maximum a posteriori estimation," in *Proc. Design Automation and Test in Europe*. Dresden, Germany, March 2014, pp. 1–6.

[5] L. Yu, L. Wei, D. A. Antoniadis, I. M. Elfadel, and D. S. Boning, "Statistical modeling with the virtual source MOSFET model," in *Proc. Design Automation and Test in Europe*. Grenoble, France, March 2013, pp. 1454–1457.

[6] L. Yu, W.-Y. Chang, K. Zuo, J. Wang, D. Yu, and D. S. Boning, "Methodology for analysis of TSV stress induced transistor variation and circuit performance," in *Proc. Int. Symp. Quality Electronic Design*. Santa Clara, CA, March 2012, pp. 216–222.

[7] R. Ghanem and P. Spanos, *Stochastic finite elements: a spectral approach*. Springer-Verlag, 1991.

[8] D. Xiu and G. E. Karniadakis, "The Wiener-Askey polynomial chaos for stochastic differential equations," *SIAM J. Sci. Comp.*, vol. 24, no. 2, pp. 619–644, Feb 2002.

[9] D. Xiu and J. S. Hesthaven, "High-order collocation methods for differential equations with random inputs," *SIAM J. Sci. Comp.*, vol. 27, no. 3, pp. 1118–1139, Mar 2005.

[10] I. Babuška, F. Nobile, and R. Tempone, "A stochastic collocation method for elliptic partial differential equations with random input data," *SIAM J. Numer. Anal.*, vol. 45, no. 3, pp. 1005–1034, Mar 2007.

[11] F. Nobile, R. Tempone, and C. G. Webster, "A sparse grid stochastic collocation method for partial differential equations with random input data," *SIAM J. Numer. Anal.*, vol. 46, no. 5, pp. 2309–2345, May 2008.

[12] ——, "An anisotropic sparse grid stochastic collocation method for partial differential equations with random input data," *SIAM J. Numer. Anal.*, vol. 46, no. 5, pp. 2411–2442, May 2008.

[13] A. Singhee and R. A. Rutenbar, "Why Quasi-Monte Carlo is better than Monte Carlo or latin hypercube sampling for statistical circuit analysis," *IEEE Trans. CAD of Integr. Circuits and Syst.*, vol. 29, no. 11, pp. 1763–1776, Nov. 2010.

[14] Z. Zhang, T. A. El-Moselhy, I. M. Elfadel, and L. Daniel, "Stochastic testing method for transistor-level uncertainty quantification based on generalized polynomial chaos," *IEEE Trans. CAD of Integr. Circuits and Syst.*, vol. 32, no. 10, pp. 1533–1545, Oct 2013.

[15] Z. Zhang, T. A. El-Moselhy, P. Maffezzoni, I. M. Elfadel, and L. Daniel, "Efficient uncertainty quantification for the periodic steady state of forced and autonomous circuits," *IEEE Trans. Circuits and Systems II: Express Briefs*, vol. 60, no. 10, Oct 2013.

[16] Z. Zhang, I. M. Elfadel, and L. Daniel, "Uncertainty quantification for integrated circuits: Stochastic spectral methods," in *Proc. Int. Conf. Computer-Aided Design*. San Jose, CA, Nov 2013, pp. 803–810.

[17] K. Strunz and Q. Su, "Stochastic formulation of SPICE-type electronic circuit simulation with polynomial chaos," *ACM Trans. Modeling and Computer Simulation*, vol. 18, no. 4, pp. 15:1–15:23, Sep 2008.

[18] J. Tao, X. Zeng, W. Cai, Y. Su, D. Zhou, and C. Chiang, "Stochastic sparse-grid collocation algorithm (SSCA) for periodic steady-state analysis of nonlinear system with process variations," in *Porc. Asia and South Pacific Design Automation Conference*, 2007, pp. 474–479.

[19] P. Manfredi, D. V. Ginste, D. De Zutter, and F. Canavero, "Stochastic modeling of nonlinear circuits via SPICE-compatible spectral equivalents," *IEEE Trans. Circuits Syst. I: Regular Papers*, 2014.

[20] R. Pulch, "Modelling and simulation of autonomous oscillators with random parameters," *Mathematics and Computers in Simulation*, vol. 81, no. 6, pp. 1128–1143, Feb 2011.

[21] I. S. Stievano, P. Manfredi, and F. G. Canavero, "Stochastic analysis of multiconductor cables and interconnects," *IEEE Trans. Electromagnetic Compatibility*, vol. 53, no. 2, pp. 501–507, May 2011.

[22] D. V. Ginste, D. D. Zutter, D. Deschrijver, T. Dhaene, P. Manfredi, and F. Canavero, "Stochastic modeling-based variability analysis of on-chip interconnects," *IEEE Trans. Components, Packaging and Manufacturing Technology*, vol. 2, no. 7, pp. 1182–1192, Jul. 2012.

[23] S. Vrudhula, J. M. Wang, and P. Ghanta, "Hermite polynomial based interconnect analysis in the presence of process variations," *IEEE Trans. CAD Integr. Circuits Syst.*, vol. 25, no. 10, pp. 2001–2011, Oct. 2006.

[24] T. Moselhy and L. Daniel, "Variation-aware stochastic extraction with large parameter dimensionality: Review and comparison of state of the art intrusive and non-intrusive techniques," in *Proc. Intl. Symp. Quality Electronic Design*, Mar. 2011, pp. 14–16.

[25] ——, "Stochastic integral equation solver for efficient variation aware interconnect extraction," in *Proc. Design Auto. Conf.*, Jun. 2008, pp. 415–420.

[26] P. Sumant, H. Wu, A. Cangellaris, and N. R. Aluru, "Reduced-order models of finite element approximations of electromagnetic devices exhibiting statistical variability," *IEEE Trans. Antennas and Propagation*, vol. 60, no. 1, pp. 301–309, Jan. 2012.

[27] N. Agarwal and N. R. Aluru, "Stochastic analysis of electrostatic MEMS subjected to parameter variations," *J. Microelectromech. Syst.*, vol. 18, no. 6, pp. 1454–1468, Dec. 2009.

[28] Z. Zhang, T. A. El-Moselhy, I. M. Elfadel, and L. Daniel, "Calculation of generalized polynomial-chaos basis functions and Gauss quadrature rules in hierarchical uncertainty quantification," *IEEE Trans. CAD Integr. Circuits Syst.*, vol. 33, no. 5, pp. 728–740, May 2014.

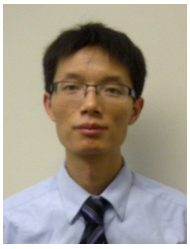
[29] L. W. T. Ng and K. E. Wilcox, "A multi-information source approach to aircraft conceptual design under uncertainty," under review.

[30] ——, "Multifidelity approaches for optimization under uncertainty," *Int. J. Numerical Meth. Eng.*, Sept. 2014.

[31] D. Allaire and K. E. Wilcox, "A mathematical and computational framework for multifidelity design and analysis with computer models," *Int. J. Uncertainty Quantification*, vol. 4, no. 1, pp. 1–20, 2014.

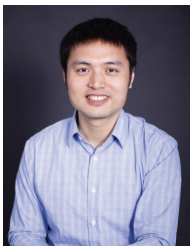
[32] E. Felt, S. Zanella, C. Guardiani, and A. Sangiovanni-Vincentelli, "Hierarchical statistical characterization of mixed-signal circuits using behavioral modeling," in *Proc. Int. Conf. Computer-Aided Design*. Washington, DC, Nov 1996, pp. 374–380.

[33] X. Yang, M. Choi, G. Lin, and G. E. Karniadakis, "Adaptive ANOVA decomposition of stochastic incompressible and compressible flows," *J. Comp. Phys.*, vol. 231, no. 4, pp. 1587–1614, Feb 2012.
[34] H. Rabitz and O. F. Alis, "General foundations of high-dimensional model representations," *J. Math. Chem.*, vol. 25, no. 2-3, pp. 197–233, 1999.
[35] M. Griebel and M. Holtz, "Dimension-wise integration of high-dimensional functions with applications to finance," *J. Complexity*, vol. 26, no. 5, pp. 455–489, Oct 2010.
[36] Z. Zhang, M. Choi, and G. E. Karniadakis, "Error estimates for the ANOVA method with polynomial choas interpolation: tensor product functions," *SIAM J. Sci. Comput.*, vol. 34, no. 2, pp. A1165–A1186, 2012.
[37] X. Ma and N. Zabaras, "An adaptive high-dimensional stochastic model representation technique for the solution of stochastic partial differential equations," *J. Compt. Phys.*, vol. 229, no. 10, pp. 3884–3915, May 2010.
[38] W. Gautschi, "On generating orthogonal polynomials," *SIAM J. Sci. Stat. Comput.*, vol. 3, no. 3, pp. 289–317, Sept. 1982.
[39] I. V. Oseledets, "Tensor-train decomposition," *SIAM J. Sci. Comput.*, vol. 33, no. 5, pp. 2295–2317, 2011.
[40] I. V. Oseledets and E. Tyrtyshnikov, "Breaking the curse of dimensionality, or how to use SVD in many dimensions," *SIAM J. Sci. Comput.*, vol. 31, no. 5, pp. 3744–3759, 2009.
[41] ——, "TT-cross approximation for multidimensional arrays," *Linear Alg. Appl.*, vol. 432, no. 1, pp. 70–88, Jan. 2010.
[42] Z. Zhang, M. Kamon, and L. Daniel, "Continuation-based pull-in and lift-off simulation algorithms for microelectromechanical devices," *J. Microelectromech. Syst.*, vol. 23, no. 5, pp. 1084–1093, Oct. 2014.
[43] G. H. Golub and J. H. Welsch, "Calculation of Gauss quadrature rules," *Math. Comp.*, vol. 23, pp. 221–230, 1969.
[44] F. L. Hitchcock, "The expression of a tensor or a polyadic as a sum of products," *J. Math. Phys.*, vol. 6, pp. 39–79, 1927.
[45] J. D. Carroll and J. J. Chang, "Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition," *Psychometrika*, vol. 35, pp. 283–319, 1970.
[46] H. Kiers, "Towards a standardized notation and terminology in multiway analysis," *J. Chemometrics*, pp. 105–122, 2000.
[47] L. R. Tucker, "Some mathematical notes on three-mode factor analysis," *Psychometrika*, vol. 31, no. 5, pp. 279–311, 1966.
[48] L. De Lathauwer, B. De Moor, and J. Vandewalle, "A multilinear singular value decomposition," *SIAM J. Matrix Anal.*, vol. 21, pp. 1253–1278, 2000.
[49] V. De Silva and L.-H. Lim, "Tensor rank and the ill-posedness of the best low-rank approximation problem," *SIAM J. Sci. Comput.*, vol. 30, no. 5, pp. 1084–1127, 2008.
[50] A. Cichoki, "Era of big data processing: A new approach via tensor networks and tensor decompositions," *arXiv Preprint, arXiv:1403.2048*, March 2014.
[51] J. Sun, D. Tao, and C. Faloutsos, "Beyond streams and graphs: Dynamic tensor analysis," in *ACM Int. Conf. Knowledge Discovery and Data Mining*, Aug. 2006, pp. 374–383.
[52] T. G. Kolda and J. Sun, "Scalable tensor decomposition for multi-aspect data mining," in *IEEE Int. Conf. Data Mining*, 2008, pp. 363–372.
[53] M. A. O. Vasilescu and D. Terzopoulos, "Multilinear analysis of image ensembles: Tensorfaces," in *Proc. Europ. Conf. Computer Vision*, 2002, pp. 447–460.
[54] D. Tao, X. Li, W. Hu, S. Maybank, and X. Wu, "Supervised tensor learning," in *Proc. Int. Conf. Data Mining*, 2005, pp. 447–460.
[55] A. Anandkumar, R. Ge, D. Hsu, S. M. Kakade, and M. Telgarsky, "Tensor decompositions for learning latent variable models," *arXiv Preprint, arXiv:1210.7559*, Oct 2012.
[56] A. Doostan and G. Iaccarino, "A least-square approximation of partial differential equations with high-dimensional random inputs," *J. Comp. Physcis*, vol. 228, pp. 4332–4345, 2009.
[57] A. Nouy, "Proper generalized decomposition and separated representations for the numerical solution of high dimensional stochastic problems," *Arch. Comp. Meth. Eng.*, vol. 27, no. 4, pp. 403–434, Dec 2010.
[58] A. Nouy and O. P. Le Maitre, "Generalized spectral decomposition for stochastic nonlinear problems," *J. Comp. Phys.*, vol. 228, pp. 205–235, 2009.
[59] B. N. Khoromskij and C. Schwab, "Tensor-structured Galerkin approximation of parametric and stochastic elliptic PDEs," *SIAM J. Sci. Comput*, vol. 33, no. 1, pp. 364–385, Oct 2011.
[60] V. Kazeev, M. Khammash, M. Nip, and C. Schwab, "Direct solution of the chemical master equation using quantized tensor trains," *PLOS Comp. Biology*, vol. 10, no. 3, pp. e1 003 359:1–19, March 2014.
[61] B. N. Khoromskij and I. Oseledets, "Quantics-TT collocation approximation of parameter-dependent and stochastic elliptic PDEs," *Comput. Methods in Appl. Math.*, vol. 10, no. 4, pp. 376–394, Jan 2010.
[62] S. Dolgov, B. N. Khoromskij, A. Litvnenko, and H. G. Matthies, "Computation of the response surface in the tensor train data format," *arXiv preprint, arXiv:1406.2816v1*, Jun 2014.
[63] D. Bigoni, A. P. Engsig-Karup, and Y. M. Marzouk, "Spectral tensor-train decomposition," *arXiv preprint, arXiv:1405.5713v1*, May 2014.
[64] I. M. Sobol, "Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates," *Math. Comp. Sim.*, vol. 55, no. 1-3, pp. 271–280, Feb 2001.
[65] P. G. Constantine, E. Dow, and Q. Wang, "Active subspace methods in theory and practice: applications to kriging surfaces," *arXiv Preprint, arXiv:1304.2070v2*, Dec. 2013.
[66] X. Li, "Finding deterministic solution from underdetermined equation: large-scale performance modeling of analog/RF circuits," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 29, no. 11, pp. 1661–1668, Nov 2011.
[67] X. Yang and G. E. Karniadakis, "Reweighted l_1 minimization method for sothcastic elliptic differential equations," *J. Comp. Phys.*, vol. 248, no. 1, pp. 87–108, Sept. 2013.
[68] J. Peng, J. Hampton, and A. Doostan, "A weighted l_1 minimization approach for sparse polynomial chaos expansion," *J. Comp. Phys.*, vol. 267, no. 1, pp. 92–111, Jun. 2014.
[69] J. Hampton and A. Doostan, "Compressive sampling of sparse polynomial chaos expansion: convergence analysis and sampling strategies," *J. Comp. Phys.*, vol. 280, no. 1, pp. 363–386, Jan. 2015.
[70] I. V. Oseledets, "TT-Toolbox 2.2," available online: `http://spring.inm.ras.ru/osel/?page_id=24`.
[71] D. R. Dereus, S. Natarajan, S. J. Cunningham, and A. S. Morris, "Tunable capacitor series/shunt design for integrated tunable wireless front end applications," in *Proc. IEEE Micro Electro Mechanical Systems*, Jan. 2011, pp. 805–808.
[72] A. K. Stamper, C. V. Jahnes, S. R. Depuis, A. Gupta, Z.-X. He, R. T. Herrin, S. E. Luce, J. Maling, D. R. Miga, W. J. Murphy, E. J. White, S. J. Cunningham, D. R. Dereus, I. Vitomirov, and A. S. Morris, "Planar MEMS RF capacitor integration," in *Proc. IEEE Solid-State Sensors, Actuators Microsyst. Conf.*, Jun. 2011, pp. 1803–1806.
[73] M. Kamon, S. Maity, D. DeReus, Z. Zhang, S. Cunningham, S. Kim, J. McKillop, A. Morris, G. Lorenz1, and L. Daniel, "New simulation and experimental methodology for analyzing pull-in and release in MEMS switches," in *Proc. IEEE Solid-State Sensors, Actuators and Microsystems Conference (TRANSDUCERS)*, Jun. 2013.
[74] "MEMS+ user's mannual," Coventor, Inc.



Zheng Zhang (S'09) received his B.Eng. degree from Huazhong University of Science and Technology, China, in 2008, and M.Phil. degree from the University of Hong Kong, Hong Kong, in 2010. Currently, he is a Ph.D student in Electrical Engineering and Computer Science at the Massachusetts Institute of Technology (MIT), Cambridge, MA. His research interests include uncertainty quantification and tensor analysis, with applications in integrated circuits (ICs), microelectromechanical systems (MEMS), power systems, silicon photonics and other emerging engineering problems.

Mr. Zhang received the 2014 IEEE Transactions on CAD of Integrated Circuits and Systems best paper award, the 2011 Li Ka Shing Prize (university best M.Phil/Ph.D thesis award) from the University of Hong Kong, and the 2010 Mathworks Fellowship from MIT. Since 2011, he has been collaborating with Coventor Inc., working on numerical methods for MEMS simulation.



Xiu Yang received his B.S. and M.S. degrees in applied mathematics from Peking University, China in 2005 and 2008, respectively, and his Ph.D. degree in applied mathematics in 2014 from Brown University, Providence, RI.

He is a postdoc research associate with the Pacific Northwest National Laboratory, Richland, WA. His research interests include uncertainty quantification, sensitivity analysis, rare events and model calibration with application to multiscale modeling, computational fluid dynamics and electronic engineering.

Ivan Oseledets received his PhD and Doctor of Sciences (second Russian degree) degrees in 2007 and 2012, respectively, both from the Institute of Numerical Mathematics of the Russian Academy of Sciences (INM RAS) in Moscow, Russia. He has worked in the INM RAS since 2003, where he is now a Leading Researcher. Since 2013 he has been an Associate Professor in Skolkovo Institute of Science and Technology (Skoltech) in Russia.

His research interests include numerical analysis, linear algebra, tensor methods, high-dimensional problems, quantum chemistry, stochastic PDEs, wavelets, data mining. Applications of interest include solution of integral and differential equations on fine grids, construction of reduced-order models for multi-parametric systems in engineering, uncertainty quantification, *ab initio* computations in quantum chemistry and material design, data mining and compression.

Dr. Oseledets received the medal of Russian Academy of Sciences for the best student work in Mathematics in 2005; the medal of Russian Academy of Sciences for the best work among young mathematicians in 2009. He is the winner of the Dynasty Foundation contest among young mathematicians in Russia in 2012.

George Karniadakis received his S.M. (1984) and Ph.D. (1987) from Massachusetts Institute of Technology (MIT). Currently, he is a Full Professor of Applied Mathematics in the Center for Fluid Mechanics at Brown University, Providence, RI. He has been a Visiting Professor and Senior Lecturer of Ocean/Mechanical Engineering at MIT since 2000. His research interests include diverse topics in computational science and engineering, with current focus on stochastic simulation (uncertainty quantification and beyond), fractional PDEs, multiscale modeling of physical and biological systems.

Prof. Karniadakis is a Fellow of the Society for Industrial and Applied Mathematics (SIAM), Fellow of the American Physical Society (APS), Fellow of the American Society of Mechanical Engineers (ASME) and Associate Fellow of the American Institute of Aeronautics and Astronautics (AIAA). He received the CFD award (2007) and the J Tinsley Oden Medal (2013) by the US Association in Computational Mechanics.

Luca Daniel (S'98-M'03) received the Ph.D. degree in electrical engineering and computer science from the University of California, Berkeley, in 2003.

He is currently an Associate Professor in the Electrical Engineering and Computer Science Department of the Massachusetts Institute of Technology (MIT). His research interests include development of integral equation solvers for very large complex systems, uncertainty quantification and stochastic solvers for large number of uncertainties, and automatic generation of parameterized stable compact models for linear and nonlinear dynamical systems. Applications of interest include simulation, modeling and optimization for mixed-signal/RF/mm-wave circuits, power electronics, MEMs, nanotechnologies, materials, Magnetic Resonance Imaging Scanners, and the human cardiovascular system.

Prof. Daniel has received the 1999 IEEE Trans. on Power Electronics best paper award; the 2003 best PhD thesis awards from both the Electrical Engineering and the Applied Math departments at UC Berkeley; the 2003 ACM Outstanding Ph.D. Dissertation Award in Electronic Design Automation; 5 best paper awards in international conferences and 9 additional nominations; the 2009 IBM Corporation Faculty Award; the 2010 IEEE Early Career Award in Electronic Design Automation; and the 2014 IEEE Trans. On Computer Aided Design best paper award.