SAND2017-5298C

Gradient-Enhanced Polynomial Chaos Methods for Circuit Simulation

Eric R. Keiter[1], Laura P. Swiler[2], and Ian Z. Wilcox[3]

1 Introduction

Sensitivity analysis and uncertainty quantification (UQ) are important capabilities for circuit simulation. In this paper, sensitivities refer to the derivatives of an objective function with respect to parameters. These parameter sensitivities give a local indication of the important parameters governing a response at a particular point. UQ allows one to understand the probability distribution of the response, given probability distributions on the input parameters.

Sampling methods are commonly used to perform UQ. While sampling is an attractive approach for several reasons (e.g it is repeatable given a particular seed, it is fault tolerant in the sense one can drop failed sample evaluations, and it is easy to understand), sampling suffers from the curse of dimensionality. A large number of samples are required to estimate the output statistics, especially to resolve small tail probabilities. The accuracy of the mean estimate obtained form a set of random samples exhibits $1/\sqrt{N}$ convergence, meaning that on average one needs to quadruple the number of sample points N to halve the error. Although many improvements on sampling schemes have been developed to overcome these limitations, such as Latin Hypercube Sampling [1, 2] and space-filling designs, the essential limitations of sampling still remain.

[1] Electrical Models and Simulation, Sandia National Laboratories, MS 1177, Albuquerque, NM, 87185 USA erkeite@sandia.gov
[2] Optimization and Uncertainty Quantification, Sandia National Laboratories, MS 1318, Albuquerque, NM, 87185 USA lpswile@sandia.gov
[3] Component and Systems Analysis, Sandia National Laboratories, MS 1177, Albuquerque, NM, 87185 USA iwilcox@sandia.gov
This work was sponsored by the Laboratory Directed Research and Development (LDRD) Program at Sandia National Laboratories. Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energys National Nuclear Security Administration under contract DE-NA0003525.

A recent interest in the computational simulation community is the use of more "embedded" UQ methods, in which the UQ algorithm is intrusively built into the simulator. As an example, in reference [3], an intrusive Galerkin based polynomial chaos expansion (PCE) method was demonstrated in a circuit simulator. However, the implementation required heavy instrumentation of the device models, which would be impractical in most production simulators.

There are categories of UQ method which require some simulator modification, but for which the required modifications are less burdensome than those necessary for fully intrusive Galerkin-based PCE. Specifically, if a simulator has been instrumented to efficiently produce parameter sensitivities [4–6], these can be used to enhance both the accuracy and runtime of several nominally non-intrusive UQ methods [7].

We outline the formulations for this UQ method, and demonstrate the computational savings that can be gained when using accurate sensitivities from an application code in the UQ process. The approaches and algorithms described in this paper are in implemented in two software frameworks: Xyce [8], a parallel circuit simulator developed at Sandia National Laboratories, and Dakota [9], an optimization and UQ toolkit also developed at Sandia. Both are open-source software packages available at https://info.sandia.gov/xyce and https://dakota.sandia.gov, respectively. However, it should be emphasized that the algorithms and approaches presented here are general, and applicable in other computational domains.

2 Transient Sensitivities

Many UQ techniques can be enhanced if the application code is able to produce parameter sensitivities with respect to objective functions of interest. In this paper, a high-level overview of direct and adjoint transient sensitivities are given. For a more detailed description the reader is encouraged to look at references [4–6]. For this work, our interest is in transient dynamical systems represented by the differential-algebraic equation (DAE) form:

$$F(x,t,p) = \frac{dq(x(t,p),p)}{dt} + j(x(t,p),p) - b(t,p) = 0, \tag{1}$$

where $x \in \mathbb{R}^{n_x}$ is the DAE solution, which will satisfy $F = 0$ for all p. In circuit simulation, x consists of nodal voltages and branch currents. $p \in \mathbb{R}^{n_p}$ is a set of input parameters. q and j are functions representing the dynamic and static circuit elements respectively, and $b(t) \in \mathbb{R}^{n_x}$ is the input vector. In circuit analysis, q mostly contains capacitor charges, j contains resistance terms and b contains independent current and voltage sources. As such q, j and b are populated by the various circuit element models (also referred to as "compact models") supported by the circuit simulator. Transient analysis of Eq. (1) requires an implicit time integration method such as Backward Euler (BE) or the trapezoid rule. $F \in \mathbb{R}^{n_x}$ is the residual equation vector that is minimized by Newton's method at each time step to solve for x.

We are also interested in objective functions of the dynamical system, $O(x,p) \in \mathbb{R}^{n_O}$. For circuit simulation, the objective function could be a circuit output voltage, or something more complex, such as a signal delay. A sensitivity is the derivative of O with respect to p, which can be expressed using the chain rule giving:

$$\frac{dO}{dp} = -\frac{\partial O}{\partial x}\left(\frac{\partial F}{\partial x}\right)^{-1}\frac{\partial F}{\partial p} + \frac{\partial O}{\partial p}, \tag{2}$$

where x and F have the same meaning as in Eq. (1). The right-hand side of Eq. (2) contains the product of several matrices, which each have different dimensions. $\partial O/\partial x$ is of dimension $n_O \times n_x$. The Jacobian matrix $\partial F/\partial x$ is of dimension $n_x \times n_x$, and will generally be available in any simulator that solves Eq. (1) using implicit methods. The derivative vector $\partial F/\partial p$ is referred to as the "function derivative", is of dimension $n_x \times n_p$, and must be populated by the various compact device models. In modern circuit simulators, with complicated device compact models, computing $\partial F/\partial p$ can be challenging and may only be practical with automatic differentiation (AD). For this Xyce uses the Sacado AD library [10].

Sensitivities can be computed using two different methods; the direct method and the adjoint method. The difference between direct and adjoint is related to the order in which the terms of Eq. (2) are computed. For problems with large numbers of parameters n_p, and a small number of objectives n_O, the adjoint method is usually more efficient. For the opposite case, the direct method is a better choice. Transient direct and adjoint sensitivities are briefly described in Sections 2.1 and 2.2 respectively.

2.1 Transient Direct Sensitivities

Transient direct sensitivities can be derived by following the approach described by Hocevar [4]. For any integration method, a transient direct sensitivity DAE equation can be derived by differentiating the original DAE (Eq. (1)) with respect to a parameter, p:

$$\frac{dF(x,t,p)}{dp} = \frac{d}{dp}\left[\frac{dq(x(t),p)}{dt} + j(x(t),p) - b(t,p)\right] = 0 \tag{3}$$

A numerical solution to Eq. (3) is obtained using an implicit time integration method. If using BE, the expanded direct sensitivity DAE equation is determined by substituting the BE formula for dq/dt and expanding the q and j derivatives using the chain rule (for example $dq/dp = \partial q/\partial x \cdot \partial x/\partial p + \partial q/\partial p$). This gives:

$$\overbrace{\left[\frac{1}{h_i}\frac{\partial q_i}{\partial x_i} + \frac{\partial j_i}{\partial x_i}\right]}^{\text{Jacobian}}\frac{\partial x_i}{\partial p} = -\overbrace{\left(\frac{1}{h_i}\left[\frac{\partial q_i}{\partial p} - \frac{\partial q_{i-1}}{\partial p}\right] + \frac{\partial j_i}{\partial p} - \frac{\partial b_i}{\partial p}\right)}^{\text{Function Derivative}} + \overbrace{\frac{1}{h_i}\left[\frac{\partial q_{i-1}}{\partial x_{i-1}}\right]\frac{\partial x_{i-1}}{\partial p}}^{\text{Chain Rule term}}, \tag{4}$$

where i is the time step index, and h_i is the time step size going from step $i-1$ to step i. Similar formulas can be derived for other integration methods. Eq. (4) is solved at each time step once the Newton loop for the original DAE has converged. The Jacobian matrix on the left-hand side of Eq. (4) is the same Jacobian as the one used in the original DAE solve, so it can simply be reused. Note that the "function derivative" on the right-hand side of Eq. (4) is equivalent to $\partial F/\partial p$ in Eq. (2), and the Jacobian in Eq. (4) is the equivalent to $\partial F/\partial x$ from Eq. (2).

2.2 Transient Adjoint Sensitivities

Transient adjoint sensitivities [5, 6] can be broadly classified into two categories: *discrete adjoint sensitivities* (in which one applies the adjoint operator after discretizing the direct sensitivity DAE) and *continuous adjoint sensitivities* (in which one applies the adjoint operator first, and then discretizes). For the sake of brevity, this paper describes the discrete adjoint form [5].

For the discrete transient case, it is convenient to consider the entire transient in block matrix form. If a transient simulation consists of N time points, then all the time points can be considered in a single block matrix equation:

$$\mathbf{F} = \dot{\mathbf{Q}} + \mathbf{J} - \mathbf{B} = \mathbf{0}, \tag{5}$$

where $\mathbf{F}$ is the block residual vector given by $\mathbf{F} = [F_0, F_1, ..., F_N]^T$. The other terms in the equation: $\mathbf{X}$,$\dot{\mathbf{Q}}$, $\mathbf{J}$, and $\mathbf{B}$ are block analogies of the original DAE equation terms: x, q, j, and b, respectively. For conventional time integration methods, the block Jacobian is a lower triangular block matrix:

$$\frac{\partial \mathbf{F}(\mathbf{X})}{\partial \mathbf{X}} = \begin{bmatrix} \left(\frac{\partial F_0}{\partial x_0}\right) & & & \\ \left(\frac{\partial F_1}{\partial x_0}\right) & \left(\frac{\partial F_1}{\partial x_1}\right) & & \\ \vdots & \vdots & \ddots & \\ & & & \left(\frac{\partial F_N}{\partial x_N}\right) \end{bmatrix}, \tag{6}$$

where the block linear system is:

$$\frac{\partial \mathbf{F}}{\partial \mathbf{X}} \Theta = \frac{\partial \mathbf{F}}{\partial p}, \tag{7}$$

and where $\Theta = [\Theta_0, \Theta_1, ..., \Theta_N]^T$ is the derivative of the solution $\mathbf{X} = [x_0, x_1, ..., x_N]^T$ with respect to a parameter value p. e.g., $\Theta_0 = dF_0/dp$. The block matrix is banded and upper triangular. Intuitively, solving this block linear system requires one to start with the upper left-hand corner of the matrix (at the first time point), and use forward substitution to solve the system. Doing this is analogous to integrating forward in time. For BE, the equivalent equation corresponding to block row i in Eq. (6) is:

$$\left(\frac{\partial F_i}{\partial x_i}\right)\frac{\partial x_i}{\partial p} = -\left(\frac{\partial F_i}{\partial x_{i-1}}\right)\frac{\partial x_{i-1}}{\partial p} + \frac{\partial F_i}{\partial p} \tag{8}$$

Eq. (8) is equivalent to Eq. (4), when the residual F is expanded using the BE formula. $\partial F_i/\partial x_i$ is the Jacobian, $\partial F_i/\partial p$ the function derivative and $\partial F_i/\partial x_{i-1}$ the block matrix off-diagonal, or "chain rule term".

One can obtain the discrete adjoint form by taking the transpose of the Eq (6). The resulting block Jacobian has the form of an upper triangular matrix:

$$\left(\frac{\partial \mathbf{F}(\mathbf{X})}{\partial \mathbf{X}}\right)^T = \begin{bmatrix} \left(\frac{\partial F_0}{\partial x_0}\right)^T & \left(\frac{\partial F_1}{\partial x_0}\right)^T & & \\ & \left(\frac{\partial F_1}{\partial x_1}\right)^T & & \\ & & \ddots & \vdots \\ & & & \left(\frac{\partial F_N}{\partial x_N}\right)^T \end{bmatrix}, \tag{9}$$

where the block linear system is:

$$\left(\frac{\partial \mathbf{F}}{\partial \mathbf{X}}\right)^T \Theta_k^\star = \frac{\partial O_k}{\partial \mathbf{X}}, \tag{10}$$

and where $\Theta^\star$ is often referred to as the adjoint. There is a unique adjoint solution for each time point k. Similarly, the local objective function O at each time point k is considered to be a unique objective function, so O_k, is $\frac{\partial O_k}{\partial \mathbf{X}} = \left[0,0,...,\frac{\partial O_k}{\partial x_k},...,0,0\right]^T$. The matrix in Eq. (9) is upper triangular, so the solution requires a backsolve, starting in the lower right-hand corner at the final time point. This corresponds to integrating backward in time. As with direct methods, a variety of integration methods can be used to compute $\Theta^\star$. The BE form, corresponding to a single block row of the transposed block system, is given by:

$$\left[\frac{1}{h_i}\frac{\partial q_i}{\partial x_i} + \frac{\partial j_i}{\partial x_i}\right]^T \theta_i^\star = \left[\frac{1}{h_{i+1}}\frac{\partial q_{i+1}}{\partial x_{i+1}}\right]^T \theta_{i+1}^\star + \left(\frac{\partial O}{\partial x_i}\right)^T. \tag{11}$$

Eq. (11) is evaluated in a loop stepping backward from the final time to the initial time.

Once $\Theta_k^\star$ has been computed for a specific time point k, it can be used to obtain dO_k/dp by taking the dot product with $\partial \mathbf{F}/\partial p$. In block matrix form this is given by:

$$\frac{dO_k}{dp} = \Theta_k^\star \cdot \frac{\partial \mathbf{F}}{\partial p}. \tag{12}$$

The derivative $\partial \mathbf{F}/\partial p$ is the function derivative. In the special case where $O = x$, then $\frac{\partial O_k}{\partial \mathbf{X}} = [0,0,...,1,...,0,0]^T$ and equation (12) provides dx/dp for a specific time point. If computing dx/dp for multiple time points, then a separate reverse integration and dot product evaluation is required for each one.

For transient adjoint sensitivities, it is necessary to completely solve the original DAE (Eq. (1)) for the entire time range first, before solving the adjoint equations to obtain sensitivities. Information must be saved during the forward solve in order to populate the Jacobians in Eq. (9), and the function derivatives in Eq. (12). For long transients this can require a lot of storage, a drawback of transient adjoints.

3 Polynomial Chaos Expansion Methods

Stochastic expansion UQ methods approximate the functional dependence of the simulation response on uncertain model parameters by expansion in a polynomial basis [11, 12]. The polynomials used are tailored to the characterization of the uncertain parameters. PCE is based on a multidimensional orthogonal polynomial approximation.

In PCE, the output response is modeled as a function of the input random variables using a carefully chosen set of polynomials. For example, PCE employs Hermite polynomials to model Gaussian random variables, as originally employed by Wiener [13]. Dakota implements the generalized PCE approach using the Wiener-Askey scheme [11], in which Hermite, Legendre, Laguerre, Jacobi, and generalized Laguerre orthogonal polynomials are used for modeling the effect of continuous random variables described by Gaussian, uniform, exponential, beta, and gamma probability distributions, respectively. These orthogonal polynomial selections are optimal for these distribution types since the inner product weighting function corresponds to the probability density functions for these continuous distributions.

To propagate input uncertainty through a model using PCE, Dakota performs the following steps: (1) input uncertainties are transformed to a set of uncorrelated random variables, (2) a basis such as Hermite polynomials is selected, and (3) the parameters of the functional approximation are determined. The general PCE for a response O has the form:

$$O(p) \approx \sum_{j=0}^{J} \alpha_j \Psi_j(p), \tag{13}$$

where each multivariate basis polynomial $\Psi_j(p)$ involves products of univariate polynomials that are tailored to the individual random variables. The response O is analogous to the objective function O described in Section 2, except that here the input parameters p are considered to be random variables and in Section 2 they are considered deterministic. If a total-order polynomial basis is used (e.g. a total order of 2 would involve terms whose exponents are less than or equal to 2, such as ${p_1}^2$, ${p_2}^2$, and $p_1 p_2$ but not ${p_1}^2{p_2}^2$), the total number of terms N in a PCE of arbitrary order k for a response function involving n uncertain input variables is given by: $(n+k)!/(n!k!)$. If on the other hand, an isotropic tensor product expansion is used with order k in each dimension, the number of terms is $(k+1)^n$. If the order k of the expansion captures the behavior of the true function, PCE methods will give very accurate results for the output statistics of the response.

In non-intrusive PCE, as in Dakota, simulations are used as black boxes and the calculation of the expansion coefficients α_j for response metrics of interest is based on a set of simulation response evaluations. To calculate these response PCE coefficients, two primary classes of approaches are used: spectral projection and regression. The spectral projection approach projects the response against each basis function $\Psi_j(p)$ using inner products and employs the polynomial orthogonality properties to extract each coefficient. Each inner product involves a multidimensional integral over the support range of the weighting function, which can be evaluated numerically using sampling, tensor-product quadrature, Smolyak sparse grid [14], or cubature [15] approaches. One advantage of PCE methods is their convergence rate [12]. For smooth functions (i.e., analytic, infinitely-differentiable) in L_2 (i.e., possessing finite variance), exponential convergence rates can be obtained under order refinement for integrated statistical quantities of interest such as mean and variance. A disadvantage of non-intrusive PCE methods is that they may not scale well to high dimensions. Recent research in adaptive refinement and sparse recovery methods strives to address this limitation [16].

In this work, we use regression-based PCE. Regression-based PCE approaches aim to solve the linear system:

$$\boldsymbol{\Psi}\boldsymbol{\alpha} \approx \boldsymbol{R} \tag{14}$$

for a set of PCE coefficients $\boldsymbol{\alpha}$ that best reproduce a set of response values $\boldsymbol{R}$. The regression approach finds a set of PCE coefficients α_j which best match a set of response values obtained from a sampling study (e.g. a design of computer experiments producing an unstructured grid of sample points sometimes called collocation points.) on the density function of the uncertain parameters [17]. The convergence of regression-based PCE approaches has been studied. It is possible to bound the number of samples necessary to identify the coefficients in the PCE expansion by using the bounds on the spectral radius of a random matrix consisting of the sample points [18]. Convergence analyses focus on the number of samples and sampling approaches for stable and accurate solution recovery. The concept of a coherence parameter is used, which is a bound on the realized spectral radius of $\boldsymbol{W}\boldsymbol{\Psi}$, where $\boldsymbol{W}$ is a diagonal, positive definite matrix. Solution recovery of the PCE coefficients using regression PCE can be guaranteed with a number of samples that is proportional to the coherence times logarithmic factor in J, the total number of basis polynomials. In some cases, the number of samples required to recover the PCE coefficients scales linearly or nearly-linearly with the number of basis polynomials [18].

Additional regression equations can be obtained through the use of derivative information (gradients and Hessians) from each collocation point, which can aid in scaling with respect to the number of random variables, particularly for adjoint-based derivative approaches. This idea is the main subject investigated in this paper. The derivative equations are added to the set of regression equations as follows:

$$\frac{dO(p)}{dp} \approx \sum_{j=0}^{J} \alpha_j \frac{d\Psi_j(p)}{dp}. \tag{15}$$

Eq. (15) is simply the derivative of the PCE response equation (Eq. (13)) with respect to the random variables of the UQ analysis. The left-hand side is ideally provided by sensitivity calculations performed by the simulator, such as described in Section 2.

Various methods can be employed to solve Eq. (14). The relative accuracy of each method is problem-dependent. Traditionally, the most frequently used method has been least squares regression. However when $\boldsymbol{\Psi}$ is under-determined, minimizing the residual with respect to the L_2 norm typically produces poor solutions. Compressed sensing methods have been successfully used to address this limitation [19, 20]. Such methods attempt to only identify the elements of the coefficient vector $\boldsymbol{\alpha}$ with the largest magnitude and enforce as many elements as possible to be zero. Such solutions are often called sparse solutions.

The convergence of gradient-enhanced regression PCE has been studied recently [21], where the authors show that the inclusion of derivative information and appropriate normalization will almost-surely lead to improved conditions for successful solution recovery. Reference [21] presents theoretical, probabilistic bounds regarding solution recovery for regression-based Hermite PCE with derivative information. This work suggests that adding gradients to the regression formulation will improve the solution recovery at a lower overall computational cost.

Dakota provides several algorithms that solve the regression formulations for PCE, including orthogonal matching pursuit, least angle regression (LARS), least absolute shrinkage (LASSO), basis pursuit, and a standard least squares. Typically, we recommend using least squares for over-determined systems and compressed sensing methods for under-determined systems, which is the case when the basis functions are augmented with additional basis functions representing gradient terms. Details of these methods are documented in the Linear Regression section of the Dakota Theory Manual [22].

4 Results for CMOS Inverter Circuit

In this section, we demonstrate the use of gradient-enhanced PCE methods on a five-stage CMOS inverter (Fig. 1). This circuit uses 10 instances of the BSIM6 [23] compact model, which in Xyce is instrumented with AD [10] to provide analytical parameter sensitivities (the "function derivative" term described in Section 2). The only other circuit element is a step input voltage source. The system to be solved is has 60 unknowns, most of which are nodal voltages. The dq/dx Jacobian is singular, so the system is a pure DAE system. The transistor models all include nonlinear capacitances. The capacitances from the first inverter form loops with the ideal voltage source input, meaning the circuit has a DAE index of two [24, 25].

In digital circuits signal delay is an important performance metric. Capacitive effects are significant delay contributors, and in this circuit example each inverter stage adds to signal delay primarily through the gate oxide capacitors. Gate oxide thickness (referred to here as δ) is thus a critical uncertain parameter, and is specified as a parametric input to the BSIM6. For the purposes of this study, all 5 NMOS devices are assumed to have the same δ_N and all 5 PMOS devices are assumed to have the same δ_P, giving 2 uncertain scalar parameters. We model these as Gaussian-distributed uncertainties, centered around a nominal value with a standard deviation of 10% of nominal. The means of δ_N and δ_P were 1.74E-9m and 2.34E-9m, respectively. The other non-uncertain transistor parameters we used are taken from the BSIM6 benchmark tests.

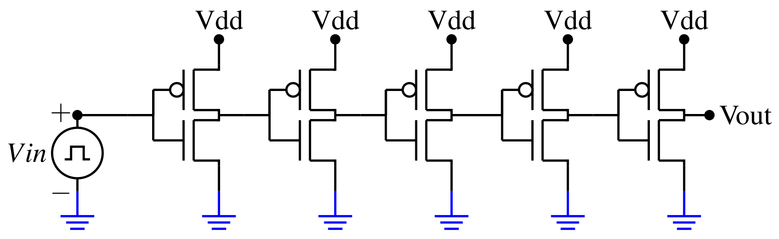

Fig. 1: CMOS circuit with five inverters.

The output of interest is the output voltage V_{out}, and a result from a forward Xyce calculation is plotted in Fig. 2. The left plot shows transient voltages for the input node, the third inverter output node, and the fifth inverter output node (V_{out}). In an ideal circuit, there would be no delay between the input and output transitions, but in this more realistic circuit that is not the case. The output voltages transition from the high state to low with some time delay after the step input, and each inverter adds additional delay to the signal. The V_{out} sensitivity with respect to δ_N and δ_P is shown on the right. Both are sharply peaked near the V_{out} transition. In this example $n_p = 2$, so the direct method (Section 2.1) was used to compute the sensitivities. However, an adjoint method (Section 2.2) produces identical results.

To quantify delay, we used a generalized Elmore delay [6] as our objective function. If $g(t) = V_{out}$ is the transient response of a node in an electrical network to a step input, the delay T_D is approximated as the centroid of its time derivative $g'(t)$:

$$T_D = \frac{\int_A^B g'(t) \cdot t \cdot dt}{\int_A^B g'(t) dt} = \frac{\int_A^B g'(t) \cdot t \cdot dt}{g(B) - g(A)}. \tag{16}$$

The parameter derivative formula for T_D is given by:

$$\frac{dT_D}{dp} = \frac{\int_A^B \frac{dg'}{dp}(t) \cdot t \cdot dt - T_D \frac{d}{dp}\left[g(B) - g(A)\right]}{g(B) - g(A)}. \tag{17}$$

The quantities T_D, $dT_D/d\delta_N$ and $dT_D/d\delta_P$ are computed with Eqs. (16) and (17) using Xyce-computed values of V_{out}, $dV_{out}/d\delta_N$ and $dV_{out}/d\delta_P$ for a sequence of time steps. The integrals are approximated numerically using trapezoid rule. The time points A and B are simply the initial and final times of the simulation.

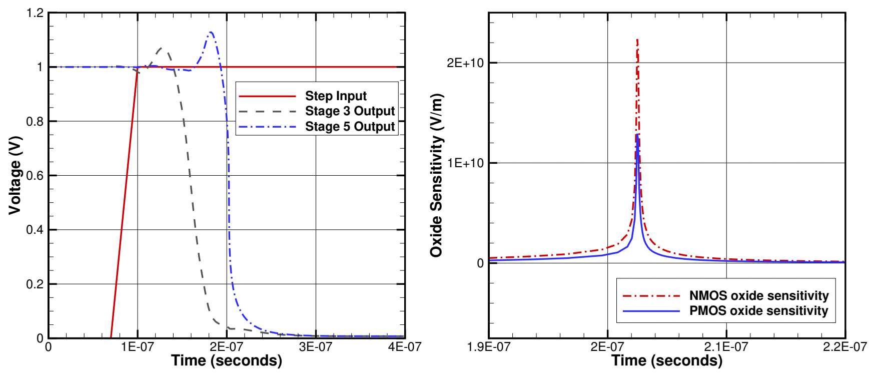

Fig. 2: Behavior of CMOS circuit exhibiting signal delay

We performed UQ on the CMOS circuit using a variety of UQ techniques. As a baseline, we performed LHS with 100 and 1000 samples. Then, we performed PCE using a full tensor product quadrature of order 5 for each of the two input parameters, requiring 25 sample points. Finally, we performed two types of regression-based PCE. In the first, we used 30 samples without gradients. In the second, we used 10 samples, where each sample included two gradient values, $dT_D/d\delta_N$ and $dT_D/d\delta_P$. Thus, the last PCE calculation used 30 pieces of information comparable to the 30 sample regression PCE with no gradients, but only required 10 samples.

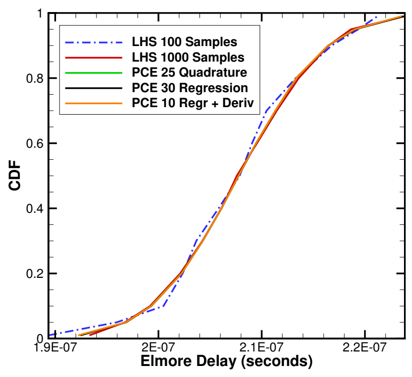

Fig. 3: CDFs for inverter delay (T_D)

Table 1: Various UQ Method Results

Number of samples and UQ method	T_D Mean	T_D Std Dev.
100 LHS	2.0781E-7	6.6309E-9
1000 LHS	2.0782E-7	6.6935E-9
25 PCE Quadrature	2.0783E-7	6.6954E-9
30 PCE Regression	2.0783E-7	6.7131E-9
10 PCE Regression with derivatives	2.0782E-7	6.7035E-9

The use of sensitivities in performing uncertainty analysis is highlighted in Fig. 3 and Table 1. As shown in the figure, the cumulative distribution function (CDF), which gives the probability that T_D is less than a particular value, is almost the same for an LHS sample of size 1000 and all of the PCE methods. The CDF curves for LHS 1000 and for all of the PCE variants overlay each other. The only one that is noticeably different is the 100 sample LHS result. Table 1 shows that the mean T_D

values are very similar, differing only in the fifth significant digit. Finally, the standard deviations show a little more variability, but again are reasonably close. We conclude that a PCE using sensitivities from Xyce (the 10 PCE regression case) performs comparably to 1000 LHS samples. Including gradients increases the cost per sample, but this additional cost is negligible for small problems. For small n_x, linear solves are less than 10% of total runtime. As a result, the two extra linear solves for each direct sensitivity time step do not incur much computational expense.

5 Conclusions

This paper explored a new approach to circuit level UQ, based on gradient-enhanced PCE. PCE is a non-sampling, projection-based technique, in which parametric uncertainties are approximated using an expansion of orthogonal polynomials. Regression-based PCE can be enhanced by parametric sensitivities from the simulator, which offers the possibility of similar accuracy with fewer samples. In this paper, transient sensitivities are described, and the successful application of these sensitivities to gradient-enhanced PCE has been demonstrated.

Acknowledgements The authors gratefully acknowledge the anonymous reviewers for their careful reading of the manuscript and their valuable suggestions to improve various aspects of this paper.

References

1. McKay, M.D., Beckman, R.J., Conover, W.J.: A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics **21**(2), 239–245 (1979)
2. Stein, M.: Large sample properties of simulations using Latin Hypercube Sampling. Technometrics **29**(2), 143–151 (1987)
3. Strunz, K., Su, Q.: Stochastic formulation of SPICE-type electronic circuit simulation with polynomial chaos. ACM Trans. Model. Comput. Simul. **18**(4), 15:1–15:23 (2008)
4. Hocevar, D.E., Yang, P., Trick, T.N., Epler, B.D.: Transient sensitivity computation for MOSFET circuits. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **CAD-4**(4), 609–620 (1985)
5. Gu, B., Gullapalli, K., Zhang, Y., Sundareswaran, S.: Faster statistical cell characterization using adjoint sensitivity analysis. In: Custom Integrated Circuits Conference, 2008. CICC 2008. IEEE, pp. 229–232 (2008)
6. Meir, A., Roychowdhury, J.: BLAST: Efficient computation of nonlinear delay sensitivities in electronic and biological networks using Barycentric Lagrange enabled transient adjoint analysis. In: DAC '12: Proceedings of the 2012 Design Automation Conference, pp. 301–310. ACM, New York, NY, USA (2012)
7. Alekseev, A.K., Navon, I.M., Zelentsov, M.E.: The estimation of functional uncertainty using polynomial chaos and adjoint equations. International Journal for Numerical Methods in Fluids **67**(3), 328–341 (2011)

8. Keiter, E.R., Aadithya, K.V., Mei, T., Russo, T.V., Schiek, R.L., Sholander, P.E., Thornquist, H.K., Verley, J.C.: Xyce Parallel Electronic Simulator: Users' Guide, Version 6.6. Tech. Rep. SAND2016-11716, Sandia National Laboratories, Albuquerque, NM (2016)
9. Adams, B.M., Bauman, L.E., Bohnhoff, W.J., Dalbey, K.R., Eddy, J.P., Ebeida, M.S., Eldred, M.S., Hough, P.D., Hu, K.T., Jakeman, J.D., Rushdi, A., Swiler, L.P., Stephens, J.A., Vigil, D.M., Wildey, T.M.: Dakota, a multilevel parallel object-oriented framework for design optimization, parameter estimation, uncertainty quantification, and sensitivity analysis: Version 6.2 users manual. Tech. Rep. SAND2014-4633, Sandia National Laboratories, Albuquerque, NM (Updated May 2015). Available online from http://dakota.sandia.gov/documentation.html
10. Phipps, E.T., Gay, D.M.: Sacado Automatic Differentiation Package. `http://trilinos.sandia.gov/packages/sacado/` (2011)
11. Xiu, D., Karniadakis, G.M.: The Wiener-Askey polynomial chaos for stochastic differential equations. SIAM J. Sci. Comput. **24**(2), 619–644 (2002)
12. Xiu, D.: Numerical Methods for Stochastic Computations: A Spectral Method Approach. Princeton University Press (2010)
13. Wiener, N.: The homogoeneous chaos. Amer. J. Math. **60**, 897–936 (1938)
14. Smolyak, S.: Quadrature and interpolation formulas for tensor products of certain classes of functions. Dokl. Akad. Nauk SSSR **4**, 240–243 (1963)
15. Stroud, A.: Approximate Calculation of Multiple Integrals. Prentice Hall (1971)
16. Constantine, P.G., Eldred, M.S., Phipps, E.T.: Sparse pseudospectral approximation method. Computer Methods in Applied Mechanics and Engineering **229–232**, 1–12 (2012)
17. Walters, R.W.: Towards stochastic fluid mechanics via polynomial chaos. In: Proceedings of the 41st AIAA Aerospace Sciences Meeting and Exhibit, AIAA-2003-0413. Reno, NV (2003)
18. Hampton, J., Doostan, A.: Coherence motivated sampling and convergence analysis of least squares polynomial chaos regression. Computer Methods in Applied Mechanics and Engineering **290**, 73–97 (2015)
19. Blatman, G., Sudret, B.: Adaptive sparse polynomial chaos expansion based on least angle regression. Journal of Computational Physics **230**(6), 2345 – 23:67 (2011)
20. Doostan, A., Owhadi, H.: A non-adapted sparse approximation of PDEs with stochastic inputs. Journal of Computational Physics **230**(8), 3015 – 3034 (2011)
21. Peng, J., Hampton, J., Doostan, A.: On polynomial chaos expansion via gradient-enhanced l1-minimization. Journal of Computational Physics **310**, 440–458 (2016)
22. Adams, B.M., Bauman, L.E., Bohnhoff, W.J., Dalbey, K.R., Eddy, J.P., Ebeida, M.S., Eldred, M.S., Hough, P.D., Hu, K.T., Jakeman, J.D., Rushdi, A., Swiler, L.P., Stephens, J.A., Vigil, D.M., Wildey, T.M.: Dakota, a multilevel parallel object-oriented framework for design optimization, parameter estimation, uncertainty quantification, and sensitivity analysis: Version 6.2 theory manual. Tech. Rep. SAND2014-4253, Sandia National Laboratories, Albuquerque, NM (Updated May 2015). Available online from http://dakota.sandia.gov/documentation.html
23. Chauhan, Y.S., Venugopalan, S., Chalkiadaki, M.A., Karim, M.A.U., Agarwal, H., Khandelwal, S., Paydavosi, N., Duarte, J.P., Enz, C.C., Niknejad, A.M., Hu, C.: BSIM6: Analog and RF compact model for bulk MOSFET. IEEE Transactions on Electron Devices **61**(2), 234–244 (2014)
24. Günther, M., Feldmann, U.: The DAE-index in electric circuit simulation. Mathematics and Computers in Simulation **39**(5-6), 573–582 (1995)
25. Bächle, S.: Index reduction for differential-algebraic equations in circuit simulation. Technical University of Berlin, Germany, Tech. Rep. MATHEON **141** (2004)