

A Non-Parametric Approach for Dynamic Range Estimation of Nonlinear Systems

Bin Wu

Jianwen Zhu

Farid N. Najm

Department of Electrical and Computer Engineering
University of Toronto, Toronto, Ontario, Canada
{bwu, jzhu, najm}@eecg.toronto.edu

ABSTRACT

It has been widely recognized that the dynamic range information of an application can be exploited to reduce the datapath bitwidth of either processors or ASICs, and therefore the overall circuit area, delay, and power consumption. Recent advances in analytical dynamic range estimation methods indicate that by systematically decomposing the system inputs into orthonormal random variables using a mathematical procedure called polynomial chaos expansion (PCE), output statistics of interest can be obtained for both linear and nonlinear systems. Despite its power for capturing both spatial and temporal correlation, the application of this method has been limited only to near-Gaussian inputs. In this paper, we propose the first algorithm with the capacity of handling both near-Gaussian and non-Gaussian input signals. Our method is based on the use of independent component analysis (ICA). Our experiments show that the new algorithm can reduce the original relative errors of 2nd order moments from 25% – 65% to 1% – 2%.

Categories and Subject Descriptors:

B.7 [Integrated Circuits]: Design Aids

General Terms: Design, Algorithms, Theory

Keywords: Dynamic range estimation, Non-Gaussian, Nonlinear, Independent component analysis, Non-parametric

1. INTRODUCTION

Today's ASIC designers start with a design specification handed off by system designers. Often in the form of C code, the algorithm-level design specification needs to be converted into register transfer level (RTL) design, typically in the form of hardware description languages. A crucial decision to be made during this process is the datapath bitwidth, including the bitwidths of different registers and functional units. An aggressively designed datapath often replaces floating-point arithmetic contained in the design specification by their fixed-point counterparts. In addition, the redundant bits that do not contribute much to the accuracy of the application are often eliminated. Such a datapath with minimal bitwidth translates to superior *circuit performance* in terms of area, speed, and power consumption. To make this possible, the dynamic range information of the application, and in the case of C code, the dynamic ranges of all declared variables and intermediate expressions (all referred to as signals or variables in the following text), have to be obtained.

Unfortunately, the common practice today for dynamic range estimation is still profiling, which works by instrumenting the original application with code that can trace the value ranges at

runtime. While this method can be very accurate, the accuracy is achieved only by extremely long and time-consuming simulation. In contrast, analytical methods can avoid long simulation by quickly analyzing the application. While many advances have been made on this front, the proposed methods have not been able to provide dynamic range information as accurate and as complete as profiling. The inaccuracy problem can usually be attributed to either no treatment of signal correlation, as in the cases of bitwidth or moment propagation method, or inadequate treatment of signal correlation, as in the case of affine arithmetic method [1]. The incompleteness problem is due to the fact that these methods typically produce only value or error bounds instead of signal distribution, thereby limiting their utility.

A new method was recently proposed [1], in which full statistics of all signals can be produced analytically and accurately. The key idea is the use of a mathematical construct, *polynomial chaos* (PC), which is a family of polynomials constructed from a set of Gaussian random variables (RVs). By decomposing an input random process into a linear combination of such polynomials, called polynomial chaos expansion (PCE), and propagating it through the system, full statistics of all signals can be easily derived. The power of the PCE method lies in its ability to capture both spatial and temporal correlation, and handle the nonlinearity of systems. However, it is limited to applications with strictly Gaussian or near-Gaussian inputs. Our study, as detailed in Section 5, indicates that that method may lead to unacceptable error for systems with non-Gaussian inputs. This seriously undermines the general applicability of the PCE method.

In this paper, we provide the first non-parametric¹ algorithm for PCE model generation. More specifically, we propose the use of a mathematical tool, independent component analysis (ICA), to facilitate PCE extraction from sample data. ICA has found its applications in neurophysiology, financial data modeling, image processing, etc. Our optimized PCE extraction method is fast and adds negligible overhead to the overall runtime.

With our proposed method, a complete PCE-based methodology for dynamic range estimation can be constructed. This methodology enjoys not only the benefits of being much faster than profiling method, and providing more detailed and accurate information than other analytical methods, as shown in [1], but also the capability of handling systems with arbitrary input characteristics. Compared with the previous PCE method, our proposed method can reduce the 2nd order moment error from 25% – 65% to 1% – 2%.

The remaining parts of this paper are organized as follows: Section 2 gives a brief introduction of polynomial chaos. Section 3, 4 describe our proposed solution. We give experimental results in Section 5 before drawing conclusion in Section 6.

2. POLYNOMIAL CHAOS

An arbitrarily distributed random variable (RV) \mathbf{x} can be expanded to a series of polynomials of Gaussian random variables, provided that $E[\mathbf{x}^2]$ (*i.e.*, the mean of \mathbf{x}^2) exists. This expansion

¹In statistics, the term *non-parametric* describes a procedure or test which may be applied irrespective of the distribution type of the underlying data

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

DAC 2005, June 13–17, 2005, Anaheim, California, USA.
Copyright 2005 ACM 1-59593-058-2/05/0006 ...\$5.00.

is referred to as polynomial chaos expansion (PCE) [1, 2]:

$$\mathbf{x} = \sum_{i=0}^{\infty} x_i \Psi_i \quad (1)$$

where x_i are constants, and Ψ_i are a set of orthogonal polynomials of *independent* standard Gaussian RVs $\{\xi_j\}$, which are called polynomial chaos (PC) [2]. Orthogonality of Ψ_i means:

$$E[\Psi_i \Psi_j] = \delta_{ij} \quad (2)$$

where δ_{ij} is Kronecker delta: $\delta_{ij} = 1$, if $i = j$; otherwise, $\delta_{ij} = 0$. Ψ_i can be constructed very easily. The degree of polynomial Ψ_i is called its polynomial chaos order, and the number of underlying independent standard Gaussian RVs ξ_j of set $\{\Psi_i\}$ are called the dimension of the polynomial chaos set. For example, for 2 dimensional PCs up to 2nd order: $\Psi_0 = 1$, $\Psi_1 = \xi_1$, $\Psi_2 = \xi_2$, $\Psi_3 = \xi_1^2 - 1$, $\Psi_4 = \xi_1 \xi_2$ and $\Psi_5 = \xi_2^2 - 1$.

In practice, (1) is truncated and only the first m terms are preserved to approximate RV \mathbf{x} .

3. BACKGROUND

The behavior of digital signal processing (DSP), as well as the behavior of general digital applications, can be captured at a high level of abstraction with a data-flow graph (DFG). Each node of the DFG represents a primitive operation, typically a multiplication or addition of real numbers, or a decision operation, as a result of a branch structure.

The output PCE of primitive operations, such as arithmetic operations, can be derived from their input PCE. For example, for addition, if $\mathbf{z} = \mathbf{x} + \mathbf{y}$, and if a PCE expansion is considered for all three variables:

$$\mathbf{x} = \sum_{i=0}^m x_i \Psi_i, \quad \mathbf{y} = \sum_{i=0}^m y_i \Psi_i, \quad \mathbf{z} = \sum_{i=0}^m z_i \Psi_i \quad (3)$$

then it is clear that:

$$z_i = x_i + y_i \quad (4)$$

More PCE propagation formulas and their derivation can be found in [1].

By PCE propagation, the PCE models for all system variables can be obtained. All statistics of the variable values are available from its PCE since PCE is a complete analytical description of the variable. For example, mean, variance, higher order moments and distribution function of variables can be easily computed from their PCEs as shown in [1].

4. PCE MODEL GENERATION BY ICA

Before we perform PCE propagation through the system, we need to build a PCE model for the system input from sample data. The PCE propagation through the system is based on this input PCE model.

Extraction of PCE model from sample data or other analytical description of random process is not a trivial problem. No previous algorithm is available except for two cases: lognormal random variable or process [3], and ideal Gaussian process [2]. The fact that no solution exists to the PCE extraction of general random process seriously restricts the applicability and accuracy of the PCE method. It is not realistic to assume that all initial inputs or parameters are Gaussian or lognormal. Our proposed PCE model generation method aims to eliminate the necessity of these oversimplified assumptions, thereby handling real world data.

In this section, we propose a novel non-parametric (*i.e.*, not dependent on a specific assumed distribution) algorithm for PCE model generation. This new method consists of three stages: a Karhunen-Loève Expansion (KLE), an Independent Component Analysis (ICA), and a Polynomial Chaos Expansion (PCE).

4.1 Karhunen-Loève Expansion

Karhunen-Loève expansion (KLE) is a well-known mathematical technique by which the zero-mean² random process $\mathbf{p}[k]$,

²we can always make a random process zero-mean by subtracting its mean function.

$k = 1, 2, \dots, N$ can be expressed as:

$$\mathbf{p}[k] = \sum_{i=1}^N \sqrt{\lambda_i} f_i[k] \mu_i \quad k = 1, 2, \dots, N \quad (5)$$

where μ_i are a set of orthonormal RVs, and λ_i and $f_i[k]$ are the eigenvalues and eigenfunctions of the autocorrelation matrix of the discrete-time random process $\mathbf{p}[k]$. The autocorrelation matrix is easy to compute either from sample traces of random process or from analytical descriptions.

The KL expansion (5) can be truncated, yielding a least-mean-squares optimal expansion on fewer variables $n < N$:

$$\mathbf{p}[k] \approx \sum_{i=1}^n \sqrt{\lambda_i} f_i[k] \mu_i \quad k = 1, 2, \dots, N \quad (6)$$

It can be shown that the relative mean square error resulting from the truncation is given by:

$$e = 1 - \frac{\sum_{i=1}^n \lambda_i}{\sum_{i=1}^N \lambda_i} \quad (7)$$

4.1.1 Independence Assumption

If $\mathbf{p}[k]$ is a Gaussian process, it can be shown that $\{\mu_i\}$ is a set of independent standard Gaussian RVs. If $\mathbf{p}[k]$ is not Gaussian, then one can only guarantee that the $\{\mu_i\}$ is an orthonormal set, that is:

$$E[\mu_i \mu_j] = \delta_{ij} \quad (8)$$

Therefore, μ_i are uncorrelated to each other, but not necessarily independent. Uncorrelatedness is weaker than independence.

For a Gaussian random process, after KLE, it is already in the required form of a PCE since the independent standard Gaussian RVs μ_i are exactly the first order polynomial chaos. No extra work is needed.

For a near-Gaussian random process, even though RVs μ_i are only uncorrelated, it is possible to approximate them as being independent and further expand every μ_i individually by PCE. The error caused by this assumption is tolerable since $\mathbf{p}[k]$ only slightly diverges from the Gaussian.

However, for a random process that is significantly non-Gaussian, the RVs μ_i cannot be treated as independent variables. Otherwise, the resultant error is large, as we will show in Section 5. In this case, we will find a linear transformation on the random vector $\boldsymbol{\mu} = [\mu_1 \mu_2 \dots \mu_n]^T$ such that the transformed RVs are mutually independent. The matrix defining this linear transformation can be found by independent component analysis (ICA), detailed in the next section.

4.2 Independent Component Analysis

Consider a random vector $\boldsymbol{\mu} = [\mu_1 \mu_2 \dots \mu_n]^T$, where the RVs $\mu_1, \mu_2, \dots, \mu_n$ are uncorrelated. ICA tries to find an invertible square matrix \mathbf{B} , so that by performing the linear transformation,

$$\mathbf{v} = \mathbf{B}\boldsymbol{\mu} \quad (9)$$

the elements ($\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$) of the resulting random vector \mathbf{v} are mutually independent or at least “as independent as possible”.

ICA can therefore be formulated as an optimization problem [4]: with an established quantitative metric for random vector independence, find the matrix \mathbf{B} , such that the metric is minimized or maximized.

4.2.1 Optimization Objective

We want to discover a matrix \mathbf{B} such that the uncorrelated RVs μ_i , forming the vector $\boldsymbol{\mu}$, can be expressed as:

$$\boldsymbol{\mu} = \mathbf{B}^{-1}\mathbf{v} \quad (10)$$

where \mathbf{v} is a vector of RVs \mathbf{v}_i that are independent (or as independent as possible). Let $\mathbf{y} = \mathbf{A}\boldsymbol{\mu}$ where \mathbf{A} is a matrix. If/when $\mathbf{A} = \mathbf{B}$, then $\mathbf{y}_i = \mathbf{v}_i$. When $\mathbf{A} \neq \mathbf{B}$, then \mathbf{y}_i is some linear combination of some of the \mathbf{v}_i 's because $\mathbf{y} = \mathbf{A}\mathbf{B}^{-1}\mathbf{v}$. The more \mathbf{v}_i 's are involved in this linear combination, the closer \mathbf{y}_i is to being Gaussian, due to the central limit theorem. Therefore, the

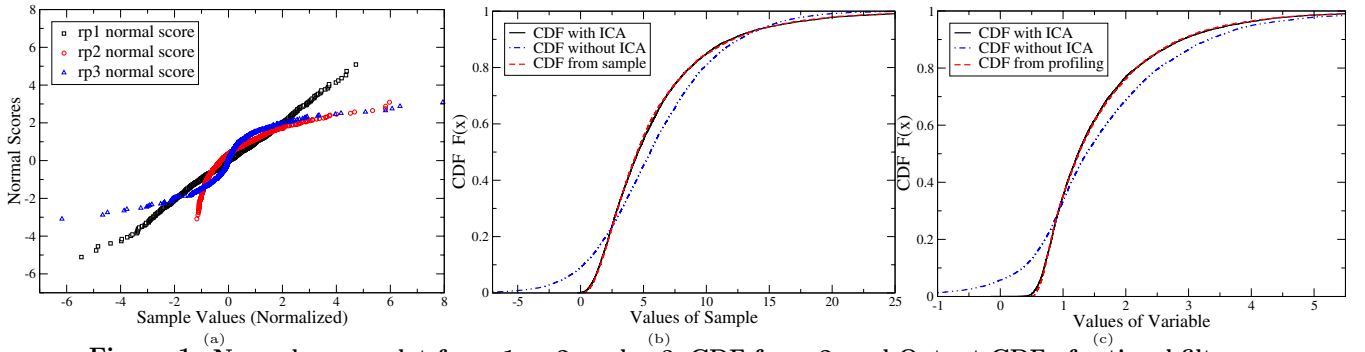


Figure 1: Normal scores plot for rp1, rp2, and rp3, CDF for rp2, and Output CDF of rational filter

case $\mathbf{y}_i = \mathbf{v}_i$ (*i.e.*, $\mathbf{A} = \mathbf{B}$) corresponds to maximizing the non-Gaussianity of \mathbf{y}_i . Thus, a row of \mathbf{B} , denoted by \mathbf{B}_i , may be obtained as the vector \mathbf{B}_i that maximizes the non-Gaussianity of the RV $\mathbf{B}_i \mathbf{v}$. This leads to the intuition that nongaussianity can serve as the optimization metric for ICA. This metric was in fact proved rigorously by information theory [4].

The natural metric for nongaussianity is entropy. However, it is difficult to compute. We instead use a simpler metric, as in [4], the absolute value of kurtosis, defined as

$$|\text{kurt}(\mathbf{v}_i)| = |E[\mathbf{v}_i^4] - 3E[\mathbf{v}_i^2]^2| \quad (11)$$

When \mathbf{v}_i is Gaussian, the above metric reaches its minimum value: 0. Therefore, maximizing nongaussianity is equivalent to maximizing the the absolute value of kurtosis.

4.2.2 Optimization Algorithm

We can determine the ICA matrix \mathbf{B} by finding its row vectors one at a time. Let \mathbf{B}_i be the i -th row vector of \mathbf{B} , then we have $\mathbf{v}_i = \mathbf{B}_i \mu$. \mathbf{B}_i is solved by maximizing the nongaussianity of individual element \mathbf{v}_i of \mathbf{v} , that is, $|\text{kurt}(\mathbf{v}_i)|$.

As a simplifying procedure, we can constrain the elements of random vector \mathbf{v} to be zero mean and unity variance, and uncorrelated to each other. This can always be achieved by preprocessing the original raw random vector by KLE first, and then only choosing \mathbf{B}_i with unity norm (*i.e.* $\|\mathbf{B}_i\| = \sqrt{\mathbf{B}_i \mathbf{B}_i^T} = 1$) to perform ICA.

Quite a few methods for ICA (also known as blind source separation) are available. In this paper, we adopt the algorithm belonging to the fixed-point category [5].

The gradient of $|\text{kurt}(\mathbf{v}_i)|$ is:

$$\frac{\partial |\text{kurt}(\mathbf{v}_i)|}{\partial \mathbf{B}_i} = \pm 4(E[\mu(\mathbf{B}_i \mu)^3])^T - 3\mathbf{B}_i \|\mathbf{B}_i\|^2 \quad (12)$$

We constrain all possible vectors \mathbf{B}_i on unity sphere by $\|\mathbf{B}_i\| = 1$. Starting with a random selected initial vector \mathbf{B}_i , conduct the iteration by replacing \mathbf{B}_i with the normalized objective gradient:

$$\mathbf{B}_i \leftarrow \frac{E[\mu(\mathbf{B}_i \mu)^3]^T - 3\mathbf{B}_i}{\|\mathbf{B}_i\|} \quad (13)$$

$$\mathbf{B}_i \leftarrow \frac{\mathbf{B}_i}{\|\mathbf{B}_i\|} \quad (14)$$

where (13) is derived from Newton iteration. Because we only care about the direction of \mathbf{B}_i , the sign part of the gradient is omitted in the above iteration. When the objective function reaches its maximum value, the direction of \mathbf{B}_i no longer changes. We terminate the iteration at this fixed point, keep the current \mathbf{B}_i and start to search for \mathbf{B}_{i+1} . In addition, at each iteration step, we need to make sure that \mathbf{B}_i is orthogonal to all previously solved \mathbf{B}_j ($j < i$) by subtracting \mathbf{B}_i 's components along all \mathbf{B}_j from \mathbf{B}_i . By doing so, we can prevent different \mathbf{B}_i from converging to the same vector and also guarantee that the elements of resultant random vectors are uncorrelated.

4.3 Expanding Independent RVs by PCE

After the ICA procedure, the random process $\mathbf{p}[k]$ is expressed in terms of a set of independent RVs \mathbf{v}_i . If we can now transform each of these RVs to a Gaussian, we would achieve a polynomial

chaos expansion (PCE) of the input process. From probability theory, we know that

$$\xi_i = \Phi^{-1}(F_i(\mathbf{v}_i)) \quad (15)$$

is a standard Gaussian RV, where $\Phi^{-1}(\cdot)$ is the inverse function of the standard Gaussian cumulative distribution function (cdf) and $F_i(\cdot)$ is the cdf of the RV \mathbf{v}_i . Notice that the cdf $F_i(\cdot)$ is *known*; it can be constructed from samples of \mathbf{v}_i , obtained from samples of μ_i via ICA, which are themselves obtained from samples of $\mathbf{p}[k]$ via KLE.

Using (15), we can construct a standard Gaussian RV ξ_i for every \mathbf{v}_i . Because RVs \mathbf{v}_i are mutually independent, RVs ξ_i are also mutually independent. We can use ξ_i as the underlying Gaussian RVs to construct polynomial chaos and expand each \mathbf{v}_i . Note that every \mathbf{v}_i corresponds to a different ξ_i . We can expand every \mathbf{v}_i individually by a 1-dimensional PCE of ξ_i since we already eliminate all dependence in sets $\{\mathbf{v}_i\}$ and $\{\xi_i\}$, that is:

$$\mathbf{v}_i = \sum_j c_{ij} \Psi_j \quad (16)$$

where Ψ_j is a polynomial in the RV ξ_i , only. If we multiply both sides of this equation by Ψ_k , and take the expectation on both sides, then by the orthogonality property of polynomial chaos, it follows that:

$$c_{ik} = \frac{E[\mathbf{v}_i \Psi_k]}{E[\Psi_k^2]} \quad (17)$$

In (17), the $E[\Psi_k^2]$ are constants which can be easily pre-computed; $E[\mathbf{v}_i \Psi_k]$ can be estimated from sample data of \mathbf{v}_i and Ψ_k ; the sample data of Ψ_k can be obtained from samples of \mathbf{v}_i by utilizing (15).

4.4 Summary

As a summary, by performing KLE, ICA and PCE extraction on the input random process $\mathbf{p}[k]$ and combining the results together we obtain the expansion of the input in PCE format:

$$\mathbf{p}[k] = \sum_i p_i[k] \Psi_i \quad (18)$$

where $p_i[k]$ is a deterministic function in discrete time domain. This PCE model will be applied at system input and propagated through the system.

Fig. 2 summarizes the complete PCE generation method. In step 9, the KLE matrix, which transforms vector μ to $\mathbf{p}[k]$ can be obtained from (6) by rewriting it in matrix format. The mean function computed in step 2 is directly passed to the final step as the 0th order PCE coefficient. Note that because the dependence among the \mathbf{v}_i 's are completely eliminated, there is no cross terms of different ξ_i in the final generated PCE model. Of course, these cross terms could be generated by nonlinear operations when input PCE passes through the system.

5. EXPERIMENTAL RESULTS

In this section, we first experimentally establish the importance of the ICA method by showing the PCE expansion error of non-Gaussian random processes when ICA is not used. We

Table 1: 1st, 2nd, and 3rd order moments: ICA vs profiling

Bench	1st order moments (mean)			2nd order moments			3rd order moments		
	Prof.	with ICA	without ICA	Prof.	with ICA	without ICA	Prof.	with ICA	without ICA
volterra	74.9	74.1(-1.03%)	74.5(-0.54%)	3.10e4	3.10e4(-0.05%)	1.84e4(-40.8%)	2.10e7	2.07e7(-1.25%)	6.21e7(-70.4%)
teager	17.0	16.8(-0.95%)	16.8(-1.23%)	1.15e4	1.16e4(0.83%)	4012(-65.1%)	5.72e6	5.63e6(-1.61%)	5.54e5(-90.3%)
adapt1	86.1	86.2(0.07%)	85.8(-0.29%)	1.05e4	1.05e4(-0.16%)	1.01e4(-3.43%)	1.63e6	1.56e6(-3.83%)	1.41e6(-13.5%)
adapt2	118	118(0.51%)	126(6.84%)	2.06e4	2.11e4(2.66%)	2.58e4(25.3%)	4.98e6	5.25e6(5.51%)	6.81e6(36.9%)
rational	1.58	1.57(-0.32%)	1.62(2.64%)	3.563	3.556(-0.2%)	4.533(27.2%)	11.6	12.0(3.82%)	16.4(41.9%)
bilinear	8.72	8.71(-0.14%)	8.80(0.92%)	140.9	137.9(-2.15%)	153.7(9.1%)	3850	3661(-4.9%)	4742(15.1%)

1. Start from sample data (traces) of discrete time random process $\mathbf{p}[k]$.
2. Compute mean function of $\mathbf{p}[k]$, and make $\mathbf{p}[k]$ zero-mean by subtracting its mean function.
3. Compute autocorrelation matrix of the resultant zero-mean process and conduct KLE on it.
4. Transform the sample traces of $\mathbf{p}[k]$ to samples of μ vector by inverse KLE matrix.
5. Compute kurtosis of μ_i , if kurtosis of all μ_i are close to 0, directly go to PCE extraction in step 8.
6. Compute ICA matrix \mathbf{B} for random vector μ as (13) and (14).
7. Use matrix \mathbf{B} to transform samples of μ to samples of \mathbf{v} .
8. Conduct PCE extraction for every \mathbf{v}_i to compute PCE coefficients c_{ij} .
9. Combine the KLE matrix, ICA matrix \mathbf{B}^{-1} and c_{ij} to form the final PCE coefficients $p_i[k]$.

Figure 2: Algorithm of PCE model generation

then demonstrate the effectiveness of ICA by measuring its accuracy for dynamic range estimation and the computational cost it incurs. Our experiments are constructed for a set of nonlinear benchmarks and random processes. All our experiments are conducted on a Sun Ultra 80 workstation.

Six benchmarks were selected from practical applications in the DSP domain. Among them, *volterra* is a second order volterra filter, which derives its name from the volterra expansion of general nonlinear functionals; *teager* implements a one-dimensional Teager’s algorithm and is commonly used for contrast enhancing in image and speech recognition; *bilinear* is a nonlinear filter whose output is linear respect to every single system variable. *Adaptive1* is a regular LMS adaptive filter which adjusts its parameters in proportion to the computed error signal, and *adaptive2* adjusts its parameters only according to the sign of the error signal. It is interesting to note that the latter includes conditionals. Finally, *rational* filter uses both nonlinear multiplications and divisions.

Sample sets of input random processes with different characteristics are generated to conduct all experiments. These processes fit the Auto-Regression Moving-Average (ARMA) model of time series, which is extensively used in engineering. In our experiment, every sample set consists of 10,000 traces. A size of such magnitude is necessary in order to make sure we do not report results containing artifacts caused by finite sample size.

5.1 Importance of ICA

We construct three random processes, one (rp1) is close to Gaussian, while the other two (rp2, rp3) are significantly non-Gaussian. Fig. 1 (a) shows normal scores plots for rp1, rp2, and rp3, respectively. The normal scores plot is a commonly used statistical technique to verify whether a set of data samples come from an underlying Gaussian distribution. If Gaussian, the data should form a straight line. It is clear that rp1 is close to Gaussian, while rp2 and rp3 significantly deviate from the Gaussian.

We use both KLE with ICA and KLE without ICA to extract PCEs from these random processes. Three cdfs are generated for every random process: two from the PCE models obtained with ICA and without ICA, another one directly from the original sample data.

For near-Gaussian process rp1, we observe that the PCE models with ICA and without ICA find similar distributions and both of them fit the original sample distribution very well. The autocorrelation functions from these three sources show the same trend. Therefore, we can conclude that for near-Gaussian cases, not performing ICA is acceptable and the method reported in [1] is sufficient.

However for significantly non-Gaussian cases, the distributions obtained with and without ICA are quite different, as shown in Fig. 1 (b). Only those obtained with ICA fit the original cdfs well. Although not shown in the paper, we observe the same for sample rp3. We can conclude that for the general case of non-Gaussian random processes, ICA is necessary. Additionally, the speed of ICA method is fast and required only a few seconds in our experiments.

5.2 Accuracy of ICA

We now demonstrate the accuracy of the ICA-based method for dynamic range estimation. We obtain our results by applying rp2 to the benchmarks. Table 1 shows the 1st order (mean), 2nd order, and 3rd order moments of benchmark outputs obtained by PCE with ICA, without ICA, and profiling respectively. We can clearly see that the statistics obtained with ICA match the profiling results very well. However the 2nd and 3rd order moments obtained without ICA have larger errors since the input random process is significantly non-Gaussian. For 2nd and 3rd order moments, the relative errors are up to 65% and 90% respectively by PCE without ICA method, while the corresponding errors are only 2% and 5% by ICA method. The errors caused by PCE without ICA is rapidly increased with moment order. To visualize the result, Fig. 1 (c) shows the distribution functions for the rational filter benchmark. It is clear that the distribution function obtained without ICA causes large errors in this case. Also in our experiment, we observe 100 – 300 speedup for PCE-based dynamic range estimation comparing with profiling.

6. CONCLUSIONS

In this paper, we propose a new non-parametric approach for dynamic range estimation, which utilizes the powerful mathematical tools of polynomial chaos expansion and independent component analysis. Our new approach can effectively eliminate errors caused by the nongaussianity of inputs and extend the previous study to both nonlinear and non-Gaussian/Gaussian cases. Its excellent accuracy and high speedup over profiling are verified by our experiments. Our PCE generation algorithm is the first algorithm in its kind with the capacity to handle general random processes (both Gaussian and non-Gaussian), and general systems (both linear and nonlinear).

7. REFERENCES

- [1] B. Wu, J. Zhu, and F. N. Najm. Dynamic range estimation for nonlinear systems. In *ACM/IEEE International Conference on Computer-Aided Design (ICCAD-04)*, San Jose, CA, November 7-11 2004.
- [2] R. G. Ghanem and P. D. Spanos. *Stochastic Finite Elements: A Spectral Approach*. Dover Publications, Inc., Mineola, NY, revised edition, 2003.
- [3] R. G. Ghanem. The nonlinear gaussian spectrum of lognormal stochastic processes and variables. *ASME Journal of Applied Mechanics*, 66(4):964–973, 1999.
- [4] A. Hyvarinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. John Wiley & Sons, New York, 2001.
- [5] A. Hyvarinen. Fast and robust fixed-point algorithms for nonlinear component analysis. *IEEE Transactions on Neural Networks*, 10(3):626–634, 1999.