Kriging Bootstrapped Neural Network Training for Fast and Accurate Process Variation Analysis

Oghenekarho Okobiah[1] and Saraju P. Mohanty[2], and Elias Kougianos[3]
NanoSystem Design Laboratory (NSDL, http://nsdl.cse.unt.edu)
University of North Texas, Denton, TX 76207, USA.
E-mail ID: oo0032@unt.edu[1], saraju.mohanty@unt.edu[2], and eliask@unt.edu[3]

***Abstract*—Speeding up the design optimization process of Analog/Mixed-Signal circuits has been a subject of active research. Techniques such as metamodeling, artificial neural networks, and optimization over SPICE netlists have been used. While the results are accurate and promising, the effects of process variation on design space exploration still persist. Metamodels created by existing techniques are still not variation aware. This paper presents a novel technique for fast and accurate process variation analysis of nanoscale circuits. The technique combines traditional Kriging with an artificial neural network (ANN) to achieve the objective. Kriging captures correlated process variations of the circuits and accurately trains the ANN to generate the metamodels. The proposed technique uses Kriging to bootstrap target samples used for the ANN training. This introduces Kriging characteristics, which account for correlation effects between design parameters, to the ANN. As a case study of the proposed method, Kriging bootstrapped trained ANN metamodels are presented for an 180 nm Phase-Locked Loop (PLL) circuit design.**

***Keywords*-Kriging, bootstrap techniques, neural networks, PLL, process variation**

I. Introduction

The development and improvement of metamodeling techniques has been gradually increasing in recent years. Significant research has been published on various metamodeling techniques for nano-CMOS applications. The goal has been to develop accurate metamodels with lower computational time costs. Extensive research work exploring polynomial, artificial neural network (ANN) and Kriging techniques have been presented in [1], [2]. ANNs are appealing because of their high accuracy and relative time efficiency. However, with the aggressive scaling of integrated circuit design, the number of design and process parameters that must be taken into consideration for design space exploration also increases. To accomplish the goal of improving the accuracy of design in the deep nanometer regions with increased number of design parameters, a particular optimization technique that has been explored is Particle Swarm Optimization (PSO) [3], [4]. PSO techniques are part of genetic, evolutionary, and population based algorithms.

While PSO and ANN metamodels have demonstrated increased accuracy [5], [6], certain design factors such as device parameter variations continue to pose a significant concern to circuit performance estimation. Analog circuits are particularly sensitive and hence prone to these effects [7].

Kriging based techniques for generating metamodels [8], [9], [10], [11] take into account the correlation between parameters and also incorporate a stochastic component that mitigates the deterministic nature of computer simulations, hence producing a more accurate statistical representation of the modeled circuit. The disadvantage of Kriging is that each point is predicted with a set of unique weights leading to time inefficient metamodel generations on large design spaces. ANN generated metamodels on the other hand are more time efficient for simulations. In this paper we propose a metamodeling based design that combines the benefits of Kriging with the accuracy and time efficiency of ANN models to produce accurate metamodels which are also more effectively process aware. Kriging is used to bootstrap the design samples used for training the ANN models, thus introducing a process aware component into the training set. We show that the Kriging trained ANN models are more process aware accurate than the bare ANN models.

The rest of this paper is organized as follows: the novel contributions of the paper are discussed in section II. A brief discussion of current related research is presented in section III. The proposed Kriging trained ANN metamodeling technique is presented in section IV. The case study circuit is described in section V. The technique of process variation aware analysis is discussed in section VI while the experimental results are presented in section VII. Finally, in section VIII conclusion and future research ideas are presented.

II. Novel Contributions

The aggressive scaling of CMOS technology for continues to drive the research for more efficient design metamodeling techniques. To this end, this paper presents the following novel-contributions to the state-of-the art:

1) The first ever method that combines Kriging and ANNs for fast and accurate metamodeling.
2) The bootstrapped Kriging accurately samples correlated data from the design space for accurate capture of process variation.
3) The Kriging-trained ANN generates very accurate metamodels which have been created with minimal effort as compared to the actual SPICE netlist.
4) A case study exploration using a 180 nm CMOS based PLL design is presented.

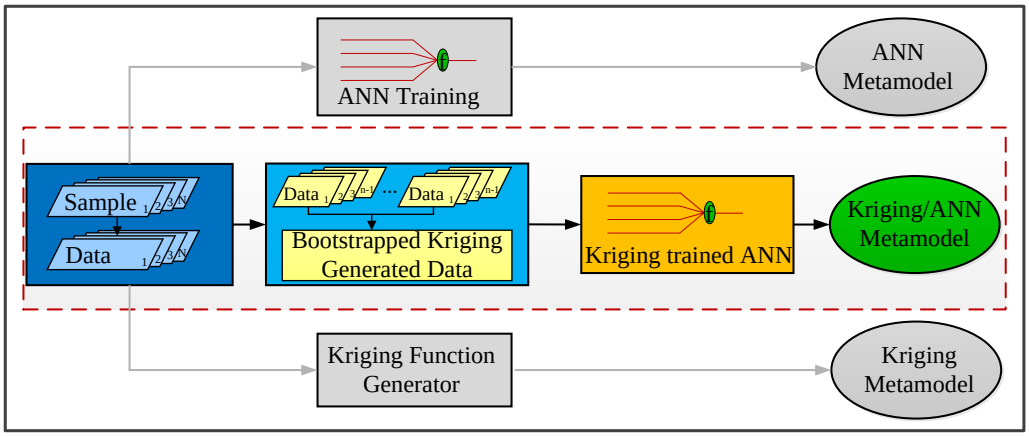


Fig. 1. Proposed Kriging Bootstrapped ANN Metamodel Generation Flow.

5) A comparative analysis of SPICE-netlist, Kriging, ANN, and Bootstrapped-Kriging-ANN approaches are presented for the 180 nm PLL.

III. Related Research

Related research to this work includes the design and formulation of modeling and metamodeling techniques. Polynomial regression methods such as response surface methodology (RSM) [12], [1], [13], [14] are one of the most common and reliable methods explored. However, low order polynomial regression techniques are not very accurate for global design space exploration [15], [16]. Non-polynomial based metamodels, particularly ANNs, have also been reported to surpass polynomial regression [17], [18], [19], [20]. ANN techniques use a learning process to continuously train weights used in approximating these models. The training process is critical in the development of ANN based models. A technique popularly used is applying optimization algorithms to optimize the weight training of ANN models [5]. It is this approach we adopt to infuse the characteristics of Kriging by bootstrapping the sample data points which are then used for the ANN training. We demonstrate that the bootstrapped data points enhance the modeling of process variation effects.

Monte Carlo (MC) simulations methods have been a reliable and effective method for yield analysis of designs. In [21], [7] hierarchical statistical analysis and regression based techniques have been explored for variation analysis. The proposed Kriging bootstrapped ANN model is analyzed for statistical variation using the MC method.

IV. Proposed Kriging Bootstrapped Artificial Neural Network (ANN) Metamodeling

In this section, we introduce and discuss the proposed Kriging bootstrapped Artificial Neural Network metamodeling technique. First we briefly introduce traditional Kriging and Artificial Neural Network metamodeling and then discuss our proposed modifications. Our proposed kriging bootstrapped Artificial Neural Network (ANN) metamodeling technique is shown in Fig. 1.

A. Kriging Based Metamodeling

Kriging prediction techniques were originally applied in geostatistics but have since been explored for other applications such as circuit design [10], [11], [9]. Kriging metamodeling combines polynomial regression with a stochastic approach to mitigate the deterministic nature of computer simulations. The Kriging equations can be expressed in the form of the following:

$$y(\mathbf{x_0}) = \sum_{j=1}^{L} \lambda_j B_j(\mathbf{x}) + z(\mathbf{x}), \tag{1}$$

where $y(\mathbf{x_0})$, is a stochastic function which predicts the response y at the design point $(\mathbf{x_0})$. $\{B_j(\mathbf{x}), j = 1, \cdots, L\}$ is a specific set of basis functions over the design domain D_N and λ_j are fitting coefficients (also known as weights) to be determined based on the Kriging method applied.

In calculating the weights λ_j for estimating Kriging functions, the autocorrelation between the input parameters is accounted for and characterized by the covariance function of the following form [22]:

$$r(\mathbf{s}, \mathbf{t}) = \mathrm{Corr}(z(\mathbf{s}), z(\mathbf{t})). \tag{2}$$

This property of Kriging prediction is exploited to model the effects of process variation on circuit metamodels. The correlations between the process variation of the design and process parameters are taken into consideration in calculating the weights for the metamodel functions. The drawback of Kriging is that the weight for each predicted point is unique and involves matrix calculations which could become time intensive for a large design space.

B. Artificial Neural Network Metamodeling

ANN models consist of simple computational elements with rich interconnections between the elements. They are modeled after biological neural networks which operate in a parallel and distributed fashion. The neural networks create models over a set of inputs by training the weights of the interconnections. Multilayer and radial neural networks are few of the commonly

employed neural networks. The multilayer network which is used in this work uses a combination of non-linear activation functions in a hidden layer and a linear activation function in the output layer. The linear layer of the function output can be expressed as follows:

$$v_i = \sum_{i=1}^{s} w_{ji} x_i + w_{j^0}, \tag{3}$$

where w_{ji} is the weight of the connection between the jth element in the hidden layer and the ith component in the input layer x_i and w_{j^0} is a constant bias [23]. The input layer is represented using a sigmoid function such as the following:

$$b_j(\nu_i) = \tanh(\lambda v_j) \tag{4}$$

The neural network utilizes an algorithm (a training function) that updates the weights and biases of the interconnections to minimize the error between the predicted point and the actual response. For this work, the ANN metamodel was created using a MATLAB toolbox which implements the Levenberg-Marquardt optimization algorithm [24].

C. Kriging Bootstrapped ANN Metamodeling

Metamodeling techniques based on Kriging prediction have been explored in [10], [13]. In estimating performance points, Kriging prediction techniques take into account the correlation effects between design parameters. This characteristic is very appealing and can be used to model the correlation effects between design parameters due to process variation for design processes deep in the nanometer range. The drawback to Kriging based techniques is that the weights used for each point prediction are unique and have to be calculated for each performance point to be estimated using linear algebra calculations (mostly matrix inversion). This can lead to potential time consuming metamodel generation for high dimensional designs and very large design spaces. Artificial Neural Network (ANN) training, which has also been presented for NanoCMOS metamodeling in [23], has been shown to be robust and accurate for high dimensional models [17]. While ANN also produces highly accurate models, it does not effectively model process variation effects with correlations present.

Hence, the proposed metamodeling technique aims to combine Kriging and ANN to generate accurate models which account for the effects of correlated process variation in a fast and efficient manner. Fig. 1 highlights the already presented methods for ANN and Kriging metamodel generation. For each method sample data points are generated using a Latin Hypercube Sampling (LHS) design and then are either fed into an ANN trainer or a Kriging function generator. In the proposed metamodel generation method, the sample data points are fed into a Kriging generator that produces an intermediate set of sample data points (bootstrapped) which are then fed into the ANN trainer. This method feeds the ANN trainer Kriging generated sample data points which are process and correlation aware. We demonstrate that using the Kriging generated sample data points will result in a more robust metamodel which is process variation aware and also less time intensive.

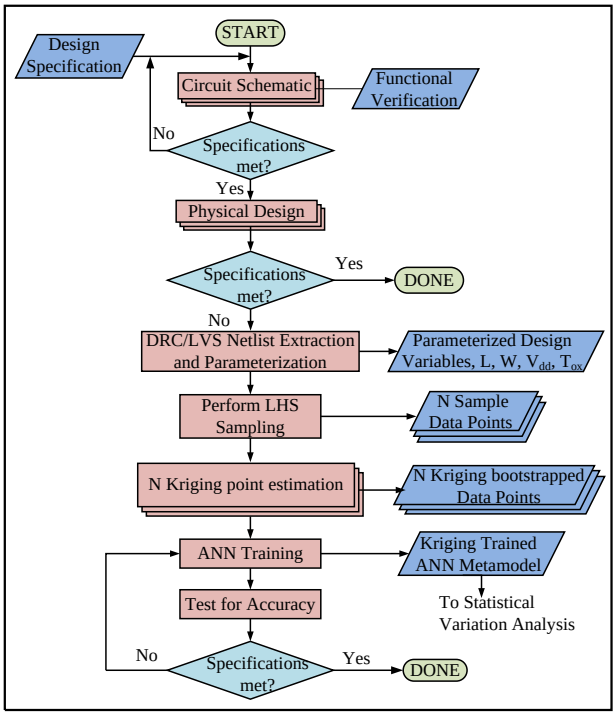


Fig. 2. Proposed Metamodel Design Flow

The methodology for the generation of the proposed metamodel-based design flow is shown in Fig. 2. The first step involves creating a SPICE netlist of the design. The functional simulation of the circuit schematic is performed to ensure the SPICE model meets design specifications. The physical layout design is also constructed using Design Rule Check (DRC) and Layout vs. Schematic (LVS) verification to ensure a match to the circuit schematic. The physical layout design is used to generate a silicon-aware accurate model (netlist). The performance of the physical design is often degraded due to the parasitic effects. A fully extracted parasitic netlist, including resistance (R), capacitance (C) and self (L) and mutual inductance (K) is used to ensure silicon-level accuracy.

The generation of the metamodel is based on the extracted parasitic $RCLK$ netlist. In order to generate data sample points, the extracted parasitic netlist is parameterized for the design and process variables and then simulated to eliminate the strenuous task of physically varying the design parameters on the physical layout design. The Latin Hypercube Sampling technique is used in the proposed method to vary the design and process parameters. LHS methods generate N random sample points from a given design space. They divide the design space into equal intervals and then randomly select design points from an interval in such a way that each interval appears once in a row-column matrix of the design space. Several techniques may be used to select the data points including uniformly, midpoints or randomly. We use Random LHS which has been reported to generate more accurate

models [1]. The LHS parameter points are used as inputs to the parameterized netlist to generate corresponding performance outputs (data point) for each sample point.

The next step in the metamodeling process is the Kriging bootstrapping of the data points. The generated sample points are fed into a Kriging metamodel generator. We implement the Kriging metamodel presented in our previous work [reference removed] for this process. We generate N Kriging bootstrapped data points by using $N-1$ points and the Kriging method to estimate the Nth point. N iterations of this process will generate N Kriging bootstrapped data points which are then used for the ANN training.

The ANN training process is used to create metamodels for each performance objective (Figure-of-Merit or FoM) characterized for the design. In this research, 4 metamodels were created for the Phase Locked Loop (PLL) circuit described in Section V.

The final step of the metamodel design flow is the verification and test of accuracy of the generated metamodel. The statistical metric used to verify the accuracy is the Root of Mean Square Error (RMSE). The expression of the RMSE is given as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(Y_i - \widehat{Y}_i\right)^2}, \tag{5}$$

where N is the number of sampled points, Y_i is the "true" circuit response (SPICE simulation results) and $\widehat{Y}_i$ is the metamodel predicted response. The RMSE measures the difference between the metamodel and the SPICE model where a smaller value indicates a more accurate model.

V. Case Study Circuit: A 180nm CMOS PLL

The phase locked loop (PLL) is a closed feedback loop circuit system whose output signal is locked to a reference input signal. The PLL is a critical component in many Analog/Mixed Signal (AMS) systems including processors, telecommunication devices,Field-Programmable Gate Arrays (FPGAs), controllers and many other systems. The system level diagram of a PLL shown in Fig. 3 shows the major components of the PLL which include the phase detector, the charge pump/loop filter, the voltage controlled oscillator (VCO) and the frequency divider.

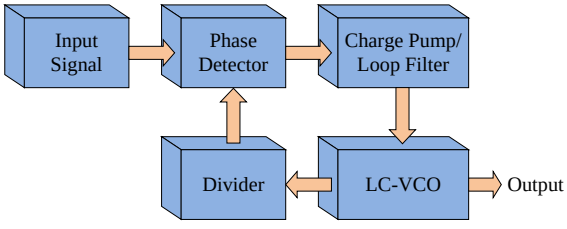


Fig. 3. High level system diagram for the PLL

The reference clock feeds the input signal to the phase detector, which compares and detects the phase difference between the input signal and the output from the VCO. The charge pump generates a supply charge in proportion to the error detected in the phase difference. The generated signal is then filtered by the loop filter to produce a control signal which the VCO uses to produce an output signal which is locked to the reference input signal. The divider is an optional component of the PLL which is used to generate an output signal which is a multiple of the reference input signal.

The schematic and physical layout design of the PLL using a 180 nm CMOS technology was produced on the CADENCE Virtuoso platform. Figure 4 shows the physical layout of the design.

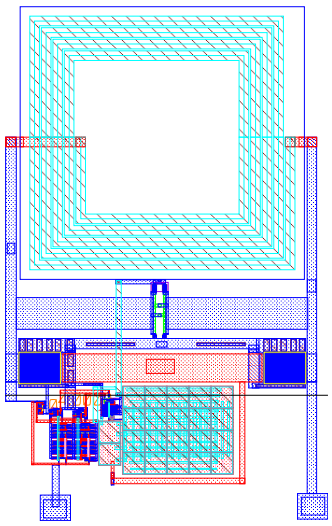


Fig. 4. Physical Layout Design for the 180 nm PLL

The PLL was characterized for power consumption, frequency output, locking time and jitter. The baseline design values are shown in Table I. The FoMs selected are Power (P_{PLL}), Frequency (F_{PLL}), Locking time (Lck_{PLL}), and Jitter (J_{PLL})). The design objective is the minimization of power consumption using the locking time as optimization cost and 21 parameters as design variables.

TABLE I
Characterization of PLL for Figures of Merit (FoM)

Circuit	P_{PLL}	F_{PLL}	Lck_{PLL}	J_{PLL}
PLL	2.48 mW	2.66 GHz	5.51 μs	16.80 ns

VI. Process Variation Aware Statistical Analysis

In this section we perform a process variation aware statistical analysis of the generated Kriging trained Neural Network metamodel. Monte Carlo simulation experiments are a common method for the analysis of process variation on analog circuits in order to estimate the yield and efficiency of the design. Monte Carlo analysis enables an efficient investigation of the design space by randomly generating a distribution test case of design variables. The set of test cases form a given probability distribution with a mean of the nominal value of the variable. This is particularly efficient

in high dimensional designs where a test case simulation time increases exponentially. For example, in our PLL case study circuit which has 21 design and process parameters, even a high and low test case will require 2^{21} simulations.

The selection of design and process parameters significantly affects the accuracy of the analysis. A sensitivity test is usually performed to select parameters which are most sensitive to performance measure. Reported research [7], [25], [26] shows that the length (L_n, L_p), width (W_n, W_p) and oxide thickness (T_{ox}) have a significant effect on the performance shift. L_n, L_p, W_n, W_p for the various sub-circuit components of the PLL have been used as design parameters. The nominal values are selected from the baseline design in section V and a Gaussian distribution with 10 % standard deviation is used to generate the sample set for the metamodel simulation. Fig. 5 summarizes the statistical analysis process. $N = 1000$ Monte Carlo simulations are performed for each FoM.

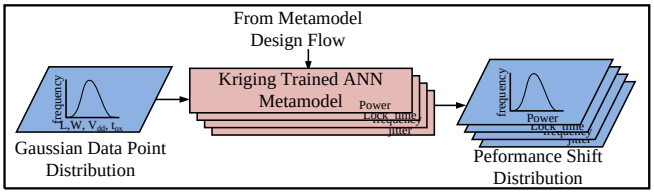


Fig. 5. Statistical Variation Analysis

The performance results of the Monte Carlo analysis are compared with an analysis from the spice simulation of the PLL design in the next section.

VII. Experimental Results

In this section, simulation experiments are performed on the case study 180 nm PLL design discussed in section V to illustrate the effectiveness of our proposed approach. The Kriging bootstrapped neural network metamodel is built using the MATLAB Neural Network toolbox and the mGstat toolbox [24], [27]. The model using the design flow discussed in Section IV-C. The extracted parasitic netlist is parameterized and used for the sample data point generation. mGstat is used to implement the Kriging boostrapping of the sample data points and then the metamodel is generated using the Neural Network toolbox. Four metamodels are generated, one for each Figure of Merit (FoM) (Power(P_{PLL}) , Frequency (F_{PLL}), Locking time (Lck_{PLL}), and Jitter (J_{PLL})) characterized for the PLL. A Monte Carlo method is used to evaluate the statistical distribution of the four FoMs. A gaussian distribution of 1000 samples is used for the simulation analysis. The results are presented in Fig. 7. Also presented in Figures 9 and 8 are statistical distributions using the artificial neural network (ANN) and the Kriging based metamodels for comparison to the proposed metamodel.

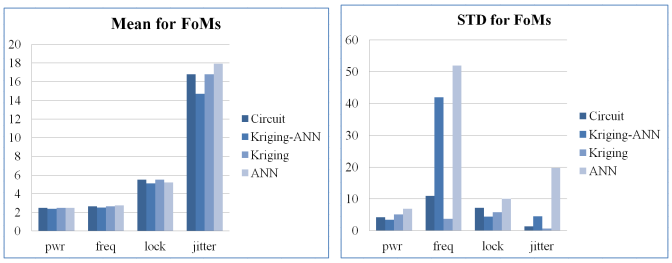
A. Results Analysis

Table II shows the accuracy of the proposed Kriging Bootstrapped Trained Neural Metamodels. The Root Mean Square Errors (RMSE) for each of the FoMs is shown. A lower value of RMSE indicates a higher accuracy. The low RMSE values thus demonstrate that the created metamodels are sufficiently accurate and can be used for design exploration.

TABLE II
Statistical Accuracy of Kriging Generated Points

FoM's	RMSE
Power (P_{PLL})	2.51 x 10 $^{-6}$
Frequency (F_{PLL})	5.68 x 10 $^{-13}$
Locking Time(Lck_{PLL})	5.01 x 10 $^{-12}$
Jitter (Lck_{PLL})	1.69 x 10 $^{-19}$

The Monte Carlo results for the various metamodels are shown in Table III. A Monte Carlo analysis on the SPICE model is used as baseline to compare the results. The results are also compared with the bare Kriging and Artificial Neural Network (ANN) metamodels.



(a) Mean (b) STD

Fig. 6. Comparative Results with Kriging and Neural Network.

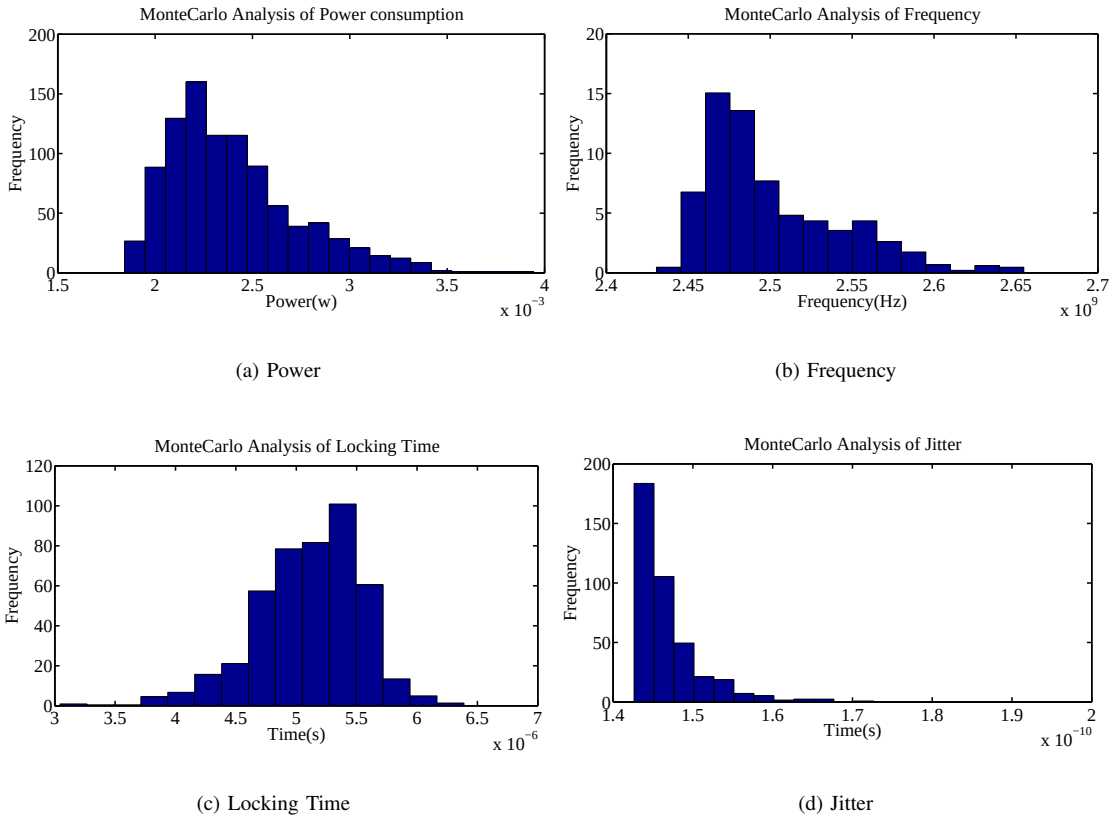
Table III shows the mean (μ) and standard deviation (σ) for the FoMs in each of the metamodels. From the results the Kriging metamodels are shown to be most accurate on both the mean (μ) and (σ) values for all FoMs. The Kriging bootstrapped neural network metamodel on the other hand is shown to be more accurate on the (σ) values than the plain neural network metamodel but less accurate on the (μ) values. This difference is expected because while bootstrapping infuses the autocorrelation property of Kriging based techniques, some error is also introduced as well. Fig. 6 shows the errors for the (μ) and (σ) as a bar chart. The histograms of the Monte Carlo analysis for the Kriging bootstrapped, Kriging and neural network metamodels are shown in Figures 7–9.

The value of the Kriging bootstrapped metamodeling technique is due to the reduced time cost for design exploration. While Kriging models may be more accurate, the time cost for design exploration for a large design space still becomes too expensive due to the repetitive solution of large-dimension systems of equations for *each* sample point. One obvious goal for metamodel use is the improved time cost. Table in IV shows the time cost for the Monte Carlo Analysis on each metamodel.

The Table shows a speedup of approximately 25 times in time cost for the Monte Carlo Simulation of 1000 runs for the Kriging bootstrapped model over traditional Kriging.

TABLE III
STATISTICAL ANALYSIS FOR ACCURACY OF NEURAL NETWORK METAMODEL FOR PLL FOMS

		Circuit	Kriging-ANN		Kriging		ANN	
		Value	Value	error (%)	Value	error (%)	Value	error (%)
P_{PLL}	Mean	2.48 mW	2.40 mW	3.22	2.50 mW	0.81	2.50 mW	0.81
	STD	0.42 mW	0.34 mW	19.05	0.51 mW	21.43	0.69 mW	64.28
F_{PLL}	Mean	2.66 GHz	2.51 GHz	5.64	2.66 GHz	0.11	2.74 GHz	5.38
	STD	10.95 MHz	41.93 MHz	282.92	3.72 MHz	66.03	51.9 MHz	373.97
Lck_{PLL}	Mean	5.51 μs	5.11μs	7.26	5.51 μs	0.07	5.20 μs	5.63
	STD	0.72 μs	0.44 μs	38.88	.58 ns	10.25	1.01 μs	40.27
J_{PLL}	Mean	16.80 ns	14.69ns	10.25	16.78ns	0.12	17.91 ns	6.61
	STD	1.32 ps	4.50 ps	240.91	0.68ps	48.48	19.17 ps	1352.22



(a) Power

(b) Frequency

(c) Locking Time

(d) Jitter

Fig. 7. Statistical Analysis of FoMs using Kriging Bootstrapped Trained Neural Network based metamodeling.

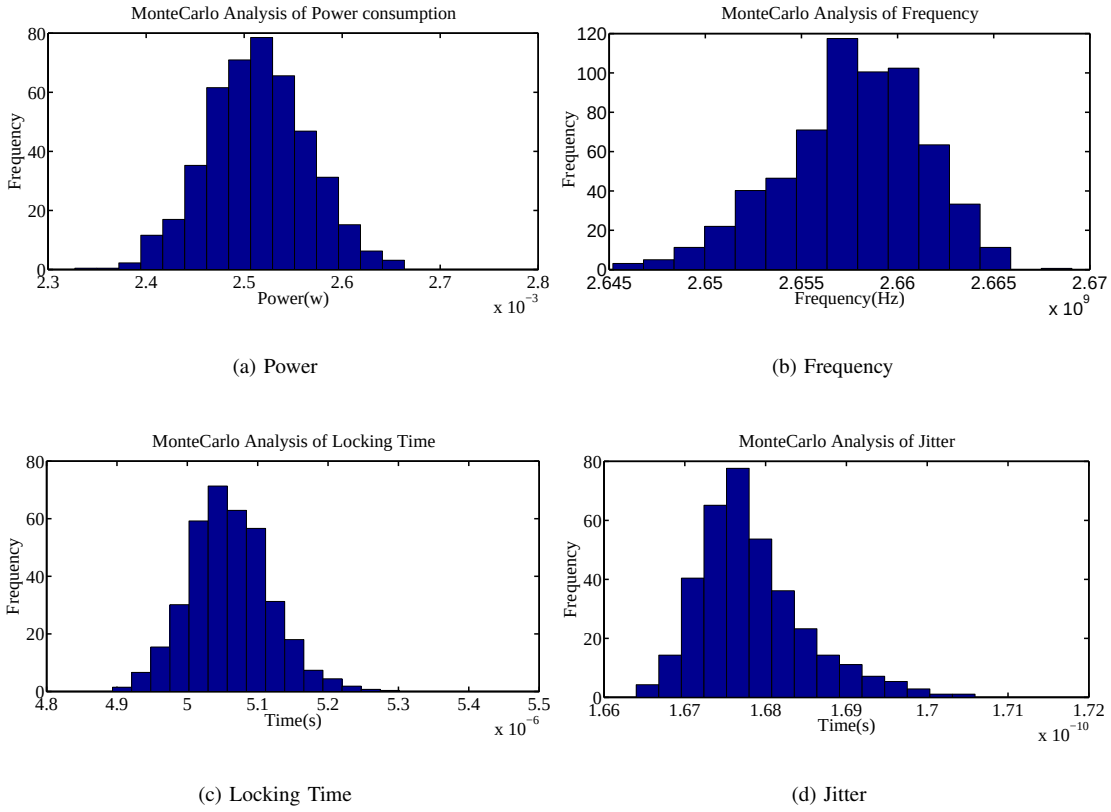
TABLE IV
MONTE CARLO TIME ANALYSIS COMPARISON FOR METAMODELS

Model	Kriging-ANN	Kriging	ANN
Time	19 s	468 s	19 s
Speedup	24.63×	1	24.63×

The significant improvement in time cost is large enough to mitigate the minimal error incurred in the metamodel. The overall use of metamodels significantly reduces the simulation time over SPICE models. It may be noted that the Monte Carlo simulation time on the SPICE models is approximately 5 days, which highlights the huge time gain with the use of metamodels.

B. Comparative Research

Table V shows a brief comparison of metamodeling based design techniques. The comparisons are only a perspective and illustrate the applicability and viability of our proposed method for statistical variability analysis. Kriging modeling is presented in [9]. In [7] a polynomial based metamodeling design including a statistical analysis on process variation is presented. A polynomial regression based technique is presented in [28]. The accuracy based on the RMSE value of the models (except for [9] which uses MSE) is shown in column 4 of Table V. The presented metamodels have been generated for different circuits, and even when the circuits are similar there are different silicon technology and performance measures making direct comparisons impossible. Hence, the comparisons are only for a broad perspective point.



(a) Power

(b) Frequency

(c) Locking Time

(d) Jitter

Fig. 8. Statistical Analysis of FoMs using Kriging based metamodeling.

TABLE V
COMPARATIVE ANALYSIS OF RELATED RESEARCH

Research	Metamodel	Test Circuit	Accuracy
Garitselov [28]	Polynomial	PLL	0.157
Yu [9]	Kriging	RO	0.5325 (MSE)
		LC-VCO	0.5325 (MSE)
Kuo [7]	Polynomial	PLL	2.0×10^{-4}
This Paper	Kriging-ANN	PLL	2.51×10^{-6}

VIII. CONCLUSION

This paper presented a metamodeling design analysis, design exploration and optimization technique that combines traditional Kriging and Artificial Neural Network to create process aware metamodels. Kriging based techniques are used to bootstrap sample data points which accommodates the correlation characteristics of Kriging techniques into the sample data. Simulation results indeed show an improved process awareness on the metamodels generated for the test case of an 180 nm Phase Locked Loop circuit. The Monte Carlo Simulation time also improved by approximately 25 times. The preliminary results are promising. Future research is planned to explore further techniques to improve the nominal accuracy of the designs.

REFERENCES

[1] O. Garitselov, S. Mohanty, and E. Kougianos, "A Comparative Study of Metamodels for Fast and Accurate Simulation of Nano-CMOS Circuits," *Semiconductor Manufacturing, IEEE Transactions on*, vol. 25, no. 1, pp. 26–36, 2012.

[2] O. Okobiah, S. P. Mohanty, E. Kougianos, and M. Poolakkaparambil, "Towards Robust Nano-CMOS Sense Amplifier Design: A Dual-Threshold versus Dual-Oxide Perspective," in *Proceedings of the 21st ACM Great Lakes Symposium on VLSI*, 2011, pp. 145–150.

[3] N. Jin and Y. Rahmat-Samii, "Advances in Particle Swarm Optimization for Antenna Designs: Real-Number, Binary, Single-Objective and Multiobjective Implementations," *Antennas and Propagation, IEEE Transactions on*, vol. 55, no. 3, pp. 556–567, 2007.

[4] C. Coello, G. Pulido, and M. Lechuga, "Handling Multiple Objectives With Particle Swarm Optimization," *Evolutionary Computation, IEEE Transactions on*, vol. 8, no. 3, pp. 256–279, 2004.

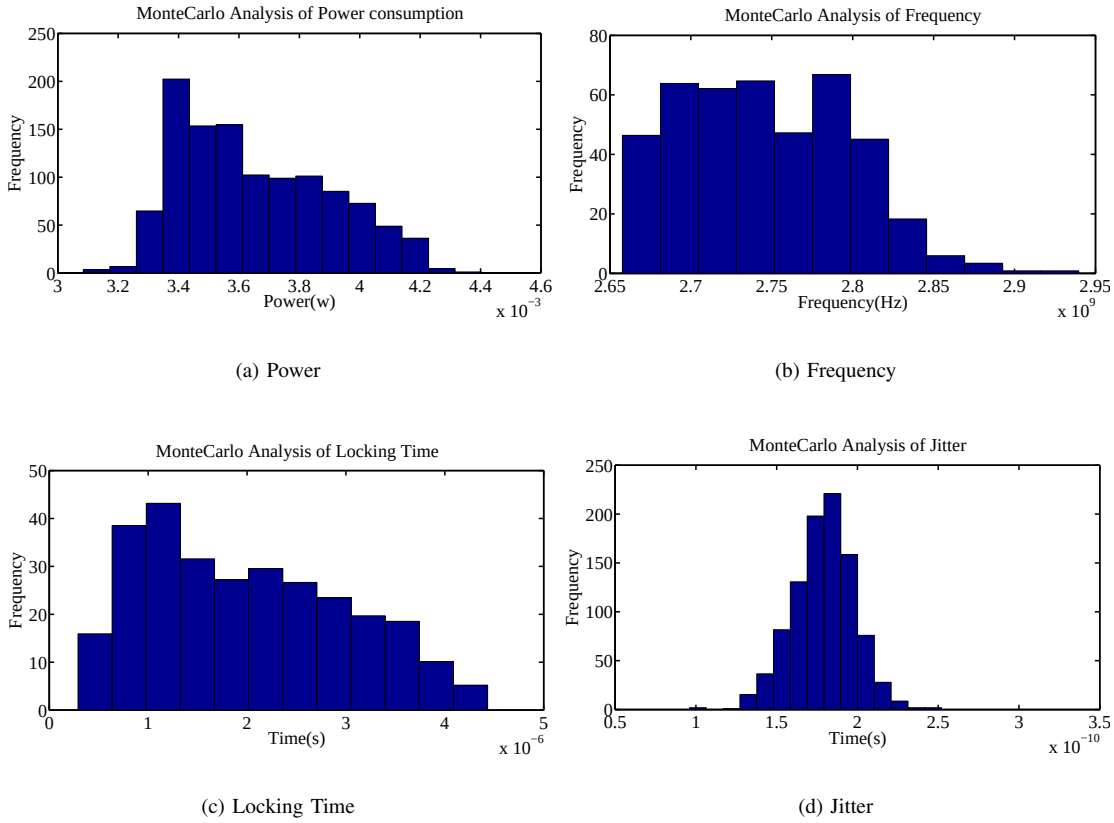
[5] I. Vilović, N. Burum, and D. Milić, "Using Particle Swarm Optimization in Training Neural Network for Indoor Field Strength Prediction," in *ELMAR, 2009. ELMAR '09. International Symposium*, 2009, pp. 275–278.

[6] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Neural Networks, 1995. Proceedings., IEEE International Conference on*, vol. 4, 1995, pp. 1942–1948.

[7] C.-C. Kuo, M.-J. Lee, C.-N. Liu, and C.-J. Huang, "Fast Statistical Analysis of Process Variation Effects Using Accurate PLL Behavioral Models," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 56, no. 6, pp. 1160–1172, June 2008.

[8] W. Van Beers, "Kriging Metamodeling in Discrete-Event Simulation: An Overview," in *Proceedings of the Winter Simulation Conference*, 2005, pp. 202–208.

[9] G. Yu and P. Li, "Yield-Aware Analog Integrated Circuit Optimization Using Geostatistics Motivated Performance Modeling," in *Computer-Aided Design, 2007. ICCAD 2007. IEEE/ACM International Conference on*, Nov. 2007, pp. 464–469.



(a) Power

(b) Frequency

(c) Locking Time

(d) Jitter

Fig. 9. Statistical Analysis of FoMs using Neural Network based metamodeling.

[10] O. Okobiah, S. P. Mohanty, E. Kougianos, and O. Garitselov, "Kriging-Assisted Ultra-Fast Simulated-Annealing Optimization of a Clamped Bitline Sense Amplifier," *VLSI Design, International Conference on*, vol. 0, pp. 310–315, 2012.

[11] H. You, M. Yang, D. Wang, and X. Jia, "Kriging Model Combined with Latin Hypercube Sampling for Surrogate Modeling of Analog Integrated Circuit Performance," in *Proceedings of the International Symposium on Quality of Electronic Design*, 2009, pp. 554–558.

[12] M. Zakerifar, W. Biles, and G. Evans, "Kriging Metamodeling in Multi-objective Simulation Optimization," in *Proceedings of the Winter Simulation Conference (WSC)*, 2009, pp. 2115–2122.

[13] W. E. Biles, J. P. C. Kleijnen, W. C. M. van Beers, and I. van Nieuwenhuyse, "Kriging Metamodeling in Constrained Simulation Optimization: An Explorative Study," in *Proceedings of the 39th Winter Simulation Conference*, 2007, pp. 355–362.

[14] G. Dellino, J. Kleijnen, and C. Meloni, "Robust Simulation-Optimization using Metamodels," in *Proceedings of the Winter Simulation Conference (WSC)*, Dec. 2009, pp. 540–550.

[15] B. Ankenman, B. Nelson, and J. Staum, "Stochastic Kriging for Simulation Metamodeling," in *Proceedings of the Winter Simulation Conference*, Dec. 2008, pp. 362–370.

[16] V. Aggarwal, "Analog Circuit Optimization using Evolutionary Algorithms and Convex Optimization," Master's thesis, Massachusetts Institute of Technology, May 2007.

[17] L. Wang, "A Hybrid Genetic Algorithm- Neural Network Strategy for Simulation Optimization," *Applied Mathematics and Computation*, vol. 170, no. 2, pp. 1329–1343, 2005.

[18] A. Khosravi, S. Nahavandi, and D. Creighton, "Developing Optimal Neural Network Metamodels Based on Prediction Intervals," in *International Joint Conference on Neural Networks*, June 2009, pp. 1583–1589.

[19] C. W. Zobel and K. B. Keeling, "Neural Network-based Simulation Metamodels for Predicting Probability Distributions," *Computers and Industrial Engineering*, vol. 54, pp. 879–888, May 2008.

[20] I. Sabuncuoglu and S. Touhami, "Simulation Metamodelling With Neural Networks: An Experimental Investigation." *International Journal of Production Research,*, vol. 40, no. 11, pp. 2483–2505, 2002.

[21] E. Felt, S. Zanella, C. Guardiani, and A. Sangiovanni-Vincentelli, "Hierarchical Statistical Characterization of Mixed-Signal Circuits Using Behavioral Modeling," in *Computer-Aided Design, 1996. ICCAD-96. Digest of Technical Papers., 1996 IEEE/ACM International Conference on*, Nov 1996, pp. 374–380.

[22] G. Bohling, "Kriging," Kansas Geological Survey, Tech. Rep., 2005.

[23] O. Garitselov, S. Mohanty, E. Kougianos, and G. Zheng, "Particle Swarm Optimization over Non-Polynomial Metamodels for Fast Process Variation Resilient Design of Nano-CMOS PLL," in *Proceedings of the great lakes symposium on VLSI*, ser. GLSVLSI '12, 2012, pp. 255–258.

[24] MATLAB, *MATLAB and Neural Network Toolbox Release 2012b*. Natick, Massachusetts, United States: The MathWorks Inc., 2012.

[25] K. Kang, B. C. Paul, and K. Roy, "Statistical Timing Analysis using Levelized Covariance Propagation," in *Proceedings Design, Automation and Test in Europe*, 2005, pp. 764–769.

[26] S. Nassif, "Modeling and analysis of manufacturing variations," in *Custom Integrated Circuits, 2001, IEEE Conference on.*, 2001, pp. 223–228.

[27] *mGstat: A Geostatistical Matlab Toolbox*. [Online]. Available: mgstat.sourcefourge.net

[28] O. Garitselov, S. Mohanty, and E. Kougianos, "Accurate Polynomial Metamodeling-Based Ultra-Fast Bee Colony Optimization of a Nano-CMOS Phase-Locked Loop," *ASP Journal of Low Power Electronics (JOLPE)*, vol. 8, no. 3, pp. 317–328, June 2012.