Efficient Decoupling Capacitance Budgeting Considering Operation and Process Variations

Yiyu Shi[†] Jinjun Xiong[*] Chunchen Liu[†] Lei He[†]
[†]Electrical Engineering Dept., UCLA [*]IBM Thomas J. Watson Research Center
[†] Los Angeles, California, 90024 [*]Yorktown Heights, New York, 10598
[†]{*yshi, lhe*}*@ee.ucla.edu* [*]*jinjun@us.ibm.com*

***Abstract*— *This paper solves the variation-aware on-chip decoupling capacitance (decap) budgeting problem. Unlike previous work assuming the worst-case current load, we develop a novel stochastic current model, which efficiently and accurately captures operation variation such as temporal correlation between clock cycles and logic-induced correlation between ports. The models also considers current variation due to process variation with spatial correlation. We then propose an iterative alternative programming algorithm to solve the decap budgeting problem under the stochastic current model. Experiments using industrial examples show that compared with the baseline model which assumes maximum currents at all ports and under the same decap area constraint, the model considering temporal correlation reduces the noise by up to* $5\times$*, and the model considering both temporal and logic-induced correlations reduces the noise by up to* $17\times$*. Compared with the model using deterministic process parameters, considering process variation (*L_{eff} *variation in this paper) reduces the mean noise by up to* $4\times$ *and the* 3σ *noise by up to* $13\times$*. While the existing stochastic optimization has been used mainly for process variation purpose, this paper to the best of our knowledge is the first in-depth study on stochastic optimization taking into account both operation and process variations for power network design. We convincingly show that considering operation variation is highly beneficial for power integrity optimization and this should be researched for optimizing signal and thermal integrity as well.*

I. Introduction

The continuous semiconductor technology scaling leads to growing process variations [1], and statistical optimization has been actively researched to cope with process variations. Recent examples include stochastic gate sizing for power reduction [2], [3] and for yield optimization [4], [5], stochastic buffer insertion to minimize clock delay [6], and adaptive body biasing with post-silicon tuning [7]. However, all these papers ignore *operation variation* such as crosstalk difference over input vectors, power supply noise fluctuation over time, and processor temperature variation over workload. We argue that a better design could be achieved by considering both operation and process variations. As a vehicle to demonstrate this point, we study in this paper the on-chip decoupling capacitance insertion and sizing (or decap budgeting) problem taking into account operation and process variations.

To solve the decap budgeting problem, most work employs a sensitivity-based optimization technique, such as those solved by either linear programming [8], quadratic programming [9], or conjugate gradient method [10], [11]. At each iteration step during optimization, sensitivities of the objective function with respect to various decaps are obtained by running circuit simulations on the adjoint network followed by time-domain convolution [9], [11]. Because both simulation and convolution are time-consuming operations, the overall runtime is high and suffers from the scalability problem for large P/G networks. To mitigate this runtime issue, different techniques have been proposed. For example, [9] employed piecewise-linear approximation for the time-domain waveforms so that convolution can be carried out faster with bounded accuracy loss. [10] exploited regular structures of P/G networks, and reduced circuit sizes by equivalent circuit transformation (such as Y-Δ transformation). Because of the reliance on special P/G structures, the applicability of this technique to large P/G networks is limited and the reduction ratio is not high in general. [11] employed a divide-and-conquer approach that partitioned a P/G network into a number of sub-circuits so that decap budgeting can be solved efficiently for each sub-circuit. But to consider the inter-dependence between different sub-circuits, an artificial boundary condition has to be imposed, hence the accuracy of the solution cannot be guaranteed. Recently, [8] used macromodeling and linear programming based approaches to solve the decap problem. However, same as the previous studies [9]–[11], it assumed a maximum current load at every port to guarantee the worst-case design scenario.

The maximum current model is over pessimistic as it ignores operation variation. Specifically, current loads at different ports are correlated and cannot reach the maximum at the same time due to the inherent logic dependency for a given design, hence exhibiting *logic-induced correlation*; and the current at a port also exhibits *temporal correlation*, i.e., the current cannot attain maximum all the time, and depending on the functionality being performed, the current variations for certain periods of clock cycles are correlated.

Unfortunately, few research has been conducted on how to extract these operation correlations. The stochastic modeling of IR drop with respect to given correlated current loads for a P/G network was studied in [12]. However, the paper did not discuss how to extract the correlation of those current loads. Moreover, it is still not clear how to use the correlation to guide the P/G network design and optimization such as decap budgeting.

In addition, the current loads are affected by process variations. [13] has considered process variation induced leakage variation for power grid analysis. While the leakage power is comparable to the dynamic power because not all components are active simultaneously in a large system-on-chip, we believe that the dynamic peak current is still dominant compared with the leakage current. However, how to design a reliable P/G network in the presence of process variation (particularly L_{eff} variation) has not been explicitly studied in existing work [9]–[11].

In this paper, we develop a novel stochastic model for current loads, taking into account operation variation such as temporal and logic-induced correlations and process variations such as systematic and random L_{eff} variation. We propose a formal method to extract operation variation and formulate a new decap budgeting problem using the stochastic current model. We develop an effective yet efficient iterative alternative programming algorithm and conduct experiments using industrial designs. We show that under the same decap area and compared with the baseline model assuming maximum currents at all ports, the model considering temporal correlation reduces the noise by up to $5\times$, and the model considering both temporal and logic-induced correlations reduces the noise by up to $17\times$. Compared with using

deterministic process parameters, considering L_{eff} variation reduces the mean noise by up to $4\times$ and the 3σ noise by up to $13\times$ when both applying the current model with temporal and logic-induced correlations. Therefore, we convincingly demonstrate the significance of considering both operation and process variations and open a new research direction for optimizing signal, power and thermal integrity with consideration of operation variation.

The remaining of the paper is organized as follows. We introduce the decap budgeting problem in Section II, and develop the stochastic current model and parameterized MNA formulation in Section III. We discuss the algorithms to solve the variation-aware decap budgeting problem in Section IV, and present experiments in Section V. We conclude in Section VI.

II. Problem Formulation

The P/G network can be modeled as a linear RLC network with each segment and pad modeled as a lumped RLC element from extraction. The behavior of any linear RLC network with p ports of interests is fully described by its state representation following the modified nodal analysis (MNA)

$$Gx + C\frac{dx}{dt} = Bu(t), \tag{1}$$

$$y = L_0^T x, \tag{2}$$

where x is a vector of nodal voltages and inductor currents, u is a vector of current sources at all ports, G is the conductance matrix, C is a matrix that includes both inductance and capacitance elements, B and L_0 are port incident matrices, and y is the output voltages of interests at the p ports.

We model the P/G network noise based upon the response $y(t)$ from (2). Because of the duality between power and ground networks, in the following, we will focus our explanation on the power network design. But it is understood that the same formulation applies to the ground network design as well. Same as [9]–[11], [14], we model the power network induced noise at a node as the integral of the voltage drop below a user specified noise ceiling $\overline{U}$ over a certain period of time:

$$z_i = \int_{\Omega_i} (\overline{U} - y_i(t))dt, \tag{3}$$

where Ω_i is the time duration when voltage at port i, y_i, drops below the noise ceiling $\overline{U}$, i.e.,

$$\Omega_i = \{t | y_i(t) \leq \overline{U}\}. \tag{4}$$

The figure of merit that measures the qualify of the whole power network design is defined as the sum of noise at all ports of interest, i.e.,

$$f = \sum_{i=1}^{p} \int_{\Omega_i} (\overline{U} - y_i(t))dt. \tag{5}$$

We will call the noise measurement in (5) simply as noise in the rest of the paper.

Based upon the noise modeling above, we can formulate the decap budgeting problem as the following optimization problem:

Formulation 1: **Decap Budgeting**: Given a power network modeled as an RLC network with specified power pads, time-varying current at different ports, and total available white space $\overline{W}$ for decoupling capacitance, the DecapOpt problem determines the places to insert decoupling capacitance and the sizes of each decoupling capacitance, such that the noise defined in (5) is minimized, considering the time-varying current $u(t)$ in (1) caused by logic-induced variation, temporal variation and process variation.

TABLE I

NOTATIONS FOR STOCHASTIC CURRENT MODEL.

Symbol	Description
p	total port number
L	max number of clock cycles for temporal correlation
$\hat{I}_k^i$	peak current at port k in clock cycle i
b_k^j	a vector of the current peaks at port k sampled every L clock cycles starting from cycle j.
$\mathcal{B}_k^j$	stochastic variable representing the samples in b_k^j
$\tilde{b}_k^j$	a vector of several b_k^j with different L_{eff}
$\tilde{\mathcal{B}}_k^j$	stochastic variable representing the samples in $\tilde{b}_k^j$
r_k	independent stochastic variables after ICA

III. Stochastic Modeling

A. Stochastic Current Modeling

In this section, we propose our stochastic current modeling for current loads of the P/G network, i.e., $u(t)$ in (1). Similar to the vectorless P/G analysis in [15], we assume that the circuit is partitioned into blocks such that different blocks are relatively independent. For each block, there are multiple ports connected to the power network, and each port is modeled as a time-varying current load for the power network. We apply extensive simulation to each block *independently* to get the current signatures. Because we ignore the interdependence between blocks, the obtained current signatures are still conservative compared with the real current profiles.

For simplicity of presentation and similar to [9] [1], we represent the current in one clock cycle as a triangular waveform with rising time, falling time, and peak value $\hat{I}$. The peak values vary in different clock cycles and over different ports. The correlation between currents for different ports is called *logic-induced correlation*. In addition, the currents of the same port in different clock cycles are also correlated. We call this type of correlation as *temporal correlation*. For example, it might take a block several clock cycles to execute certain functions and the current profile inside those clock cycles are dependent to each other. For simplicity, we assume that for a given design, the clock cycles can be divided into several trunks: the currents inside the same trunk are correlated between different clock cycles, while there is no correlation between the currents in clock cycles of different trunks. We denote L as the maximum number of clock cycles within a trunk. For example, we can choose L to be the largest number of clock cycles to finish one instruction, and will verify the validity of this choice later.

In the following, we devise a stochastic model which can efficiently capture the correlation from both the logic-induced variation and temporal variation, as well as from process variation. For simplicity of presentation, we summarize notations for the stochastic current model in Table I.

1) Stochastic Model to Consider Current Interdependence: We record the peak currents at port k ($1 \leq k \leq p$ with p as the total port number) at different clock cycles, and put them into vectors, i.e.,

$$b_k^j = [\hat{I}_k^j, \hat{I}_k^{j+L}, \hat{I}_k^{j+2L}, \ldots], \quad 1 \leq k \leq p, 1 \leq j \leq L \tag{6}$$

where $\hat{I}_k^j$ is the peak currents at port k in clock cycle j, and b_k^j is the set of peak currents sampled every L clock cycles starting from cycle j. For example, if the peak values in each clock cycle for port 1 are $[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8]$, and for port 2 are $[0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08]$, and we choose $L = 2$,

[1] Our noise verification in the experiment part does not depend on this assumption.

then

$$b_1^1 = [0.1, 0.3, 0.5, 0.7], \quad b_2^1 = [0.01, 0.03, 0.05, 0.07],$$
$$b_1^2 = [0.2, 0.4, 0.6, 0.8], \quad b_2^2 = [0.02, 0.04, 0.06, 0.08]. \tag{7}$$

We model the peak current at each port as a stochastic process. Then all the elements of b_k^j are the samples for the stochastic variable $\mathcal{B}_k^j$ with its mean $\mu(\mathcal{B}_k^j)$ and standard deviation $\sigma(\mathcal{B}_k^j)$. We call the correlation between b_k^{j1} and b_k^{j2} as temporal correlation, and the one between $b_{k_1}^j$ and $b_{k_2}^j$ as logic-induced correlation.

With those stochastic variables B_k^j's and their corresponding samples b_k^j's, we can compute the logic-induced correlation matrix $\rho(j; k_1, k_2)$ which describes the correlation between the peak currents at any two ports k_1 and k_2 in clock cycle j as

$$\rho(j; k_1, k_2) = \frac{cov(\mathcal{B}_{k_i}^j, \mathcal{B}_{k_2}^j)}{\sigma(\mathcal{B}_{k_1}^j)\sigma(\mathcal{B}_{k_2}^j)}, \quad (1 \le k_1, k_2 \le p), \tag{8}$$

where $cov(\mathcal{B}_{k_1}^j, \mathcal{B}_{k_2}^j)$ are the covariance between $\mathcal{B}_{k_1}^j$ and $\mathcal{B}_{k_2}^j$, and $\sigma(\mathcal{B}_{k_1}^j)$ and $\sigma(\mathcal{B}_{k_2}^j)$ are their standard deviations, respectively. Similarly, the temporal correlation matrix $\rho(j_1, j_2; k)$ which describes the correlation between the peak currents between clock cycles j_1 and j_2 of a same port k can be computed as

$$\rho(j_1, j_2; k) = \frac{cov(\mathcal{B}_k^{j_1}, \mathcal{B}_k^{j_2})}{\sigma(\mathcal{B}_k^{j_1})\sigma(\mathcal{B}_k^{j_2})}, \quad (1 \le j_1, j_2 \le L). \tag{9}$$

As an example, the block-diagonal structured temporal correlation matrix in Figure 1 is extracted by our method from an industry design, where the maximum instruction period is 10 clock cycles. We extract the correlation matrix for 40 clock cycles. It can be seen that the correlation matrix can be clearly divided into four trunks, and L can be set as 10. The correlation between clock cycles in different trunks is very small. This verifies that it is reasonable to take L as the maximum number of clock cycles to finish one instruction.

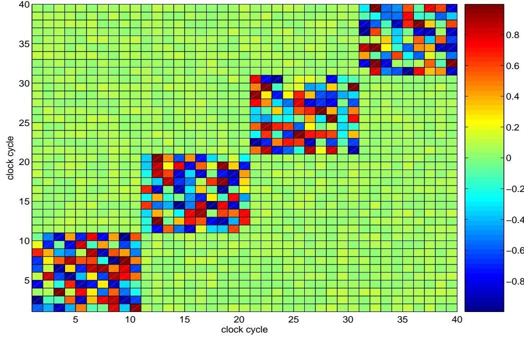

Fig. 1. The correlation map for peak currents between different clock cycles of one port from an industry application.

2) Extension to Process Variation with Spatial Correlation: [16] relates the current to the process parameters L_{eff}, t_{ox} and V_t as

$$\hat{I}_k^i \sim L_{eff}^{-0.5} t_{ox}^{-0.8} (V_{dd} - V_t). \tag{10}$$

As pointed out in [17], in 90nm regime the most significant variation source is the effective channel length (L_{eff}), and L_{eff} variation can be more than 30%. Furthermore, L_{eff} variation is mostly spatially correlated but not random [16]. Therefore, we will use L_{eff} variation as an example to show how process variation can be embedded into our stochastic modeling. It is understood that the process variation of other parameters can be dealt with in a similar way.

We use the variation model for L_{eff} based on [16]:

$$L_{eff} = L_0 + L^{prox} + L^{spat} + \epsilon, \tag{11}$$

where L_0 is the overall mean, L^{prox} is a discrete stochastic variable with a distribution determined by the frequency of each gate, L^{spat} corresponds to the spatial variation, and ϵ is the local random variation.

From (10), with L_{eff} variation, the sample $\hat{I}_k^j$ becomes a set of samples

$$\left[\hat{I}_k^j \sqrt{\frac{\bar{L}_{eff,k}}{L_{eff,k}^1}}, \hat{I}_k^j \sqrt{\frac{\bar{L}_{eff,k}}{L_{eff,k}^2}}, \dots \right], \tag{12}$$

where $L_{eff,k}^i$ with different i are the samples of $L_{eff,k}$ for the circuit block corresponding to port k with the nominal value $\bar{L}_{eff,k}$, and $\hat{I}_k^j$ are the peak current sample for $\mathcal{B}_k^j$ in the deterministic case without L_{eff} variation in (6). In other words, if we have n samples for $L_{eff,k}$ ($L_{eff,k}^1, L_{eff,k}^2, \dots, L_{eff,k}^n$), then every current sample I_k^j becomes n samples. Therefore, the sample vector b_k^j becomes n times longer in the presence of L_{eff} variation, and we denote this new vector as $\tilde{b}_k^j$. In addition, we denote the stochastic variable representing the set of $\tilde{b}_k^j$ as $\tilde{\mathcal{B}}_k^j$. In this case, the temporal correlation (8) becomes

$$\tilde{\rho}(j; k_1, k_2) = \frac{cov(\tilde{\mathcal{B}}_{k_i}^j, \tilde{\mathcal{B}}_{k_2}^j)}{\sigma(\tilde{\mathcal{B}}_{k_1}^j)\sigma(\tilde{\mathcal{B}}_{k_2}^j)}, \quad (1 \le k_1, k_2 \le p), \tag{13}$$

and the logic-induced correlation (9) becomes

$$\tilde{\rho}(j_1, j_2; k) = \frac{cov(\tilde{\mathcal{B}}_k^{j_1}, \tilde{\mathcal{B}}_k^{j_2})}{\sigma(\tilde{\mathcal{B}}_k^{j_1})\sigma(\tilde{\mathcal{B}}_k^{j_2})}, \quad (1 \le j_1, j_2 \le L). \tag{14}$$

B. Parameterized Problem Formulation

1) Parameterized Current via ICA: Directly considering the temporal and logic-induced correlation including process variation as formulated in (13) and (14) is difficult for optimization. Therefore, we propose to remove the correlation between $\tilde{B}_k^j$'s and build a parameterized current model in the following.

If all those variable $\tilde{B}_k^j$'s are Gaussian, we can apply principal component analysis (PCA) to each cluster to remove the interdependence between the stochastic variables $\tilde{\mathcal{B}}_k^j$'s. However, this is not the case for our stochastic current model. Therefore, we use independent component analysis (ICA) that is applicable to non-Gaussian distribution [18]. The input to ICA is the samples $\tilde{b}_k^j$ as well as their correlation matrices (13) and (14), and the output are a set of independent stochastic variables r_i and their corresponding coefficients $a_i(j,k)$ to reconstruct each $\tilde{\mathcal{B}}_k^j$, i.e.

$$\tilde{\mathcal{B}}_k^j = \sum_{i=1}^{q} a_i(j,k) r_i. \tag{15}$$

The order q is determined for each design such that the relative error between the original currents and model predicted currents is less than 5%. The probability density function (PDF) for each r_i is also given in the output of ICA as a one-dimensional lookup table, based on which we can bound the range of r_i as

$$\underline{r_i} \le r_i \le \overline{r_i}, \tag{16}$$

where $\underline{r_i}$ and $\overline{r_i}$ can be related to r_i's mean (μ) and variance (σ^2). For example, we can take $\underline{r_i}$ as $\mu - 4\sigma$ and $\overline{r_i}$ as $\mu + 4\sigma$.

Therefore, assuming uniform rising and falling times across the chip for the triangular current waveform within a clock cycle [2],

[2]This uniform assumption does not affect the results in our experiments.

together with $a_i(j,k)$ which represents the i-th component of the peak current at port k in clock cycle j, we have all the necessary information to obtain the i-th time-varying current waveform component $u_i(t;j,k)$. If we denote T as the clock period, then $jT \leq t \leq (j+1)T$. Put those $u_i(t;j,k)$ at all ports in clock cycle j together as

$$u_i(t;j) = \begin{pmatrix} u_i(t;j,1) \\ u_i(t;j,2) \\ \vdots \\ u_i(t;j,p) \end{pmatrix}, \quad jT \leq t \leq (j+1)T, \tag{17}$$

and then combine all the $u_i(t;j)$ in different clock cycles, we can get $u_i(t)$ with $0 \leq t \leq LT$. Finally, according to superposition theorem, we have

$$u(t) = \sum_{i=1}^{q} u_i(t) r_i, \quad 0 \leq t \leq LT. \tag{18}$$

As an illustration, Figure 2 shows the procedure for $L = 3$, $T = 1$ and $p = 1$. We call (18) as parameterized current load model.

2) Parameterized MNA for Decap Budgeting: Considering the inherent parasitics, we model the decap similarly to [19] as an equivalent series capacitor (ESC), and equivalent series resistor (ESR) and an equivalent series inductor (ESL). When a decap with size w_i is inserted into the power network at a given location, its impact can be considered by adjusting matrices G and C in (1) based on the location at the network and the size of the decap. Mathematically, it can be represented as

$$G = G_0 + \sum_{i=1}^{M} w_i \cdot G_{w,i}, \tag{19}$$

$$C = C_0 + \sum_{i=1}^{M} w_i \cdot C_{w,i}, \tag{20}$$

where G_0 and C_0 are the original matrices for the power network without decap, M is the total number of decaps, and $G_{w,i}$ and $C_{w,i}$ provide the stamping of a unit width decap at the i-th location. Due to the placement constraint, w_i has an upper bound, i.e.,

$$0 \leq w_i \leq \overline{w_i}. \tag{21}$$

The MNA equation of (1) with G given by (19), C given by (20), and u given by (18) can be written as follows

$$\begin{aligned} &(G_0 + \sum_{i=1}^{M} w_i \cdot G_{w,i})x + (C_0 + \sum_{i=1}^{M} w_i \cdot C_{w,i})\frac{dx}{dt} \\ = \; &B \sum_{i=1}^{q} u_i(t) r_i, \end{aligned} \tag{22}$$

where $0 \leq t \leq LT$ and r_i is a stochastic variable with $\underline{r_i} \leq r_i \leq \overline{r_i}$. We call this MNA equation as *parameterized MNA formulation* for decap budgeting. One of the major advantages in using this parameterized MNA formulation is that it enables us to implicitly compute sensitivities efficiently and accurately, which will become clearer in the later part of this paper.

With the parameterized MNA, the variation-aware decap budgeting problem can be mathematically represented as follows:

$$\begin{aligned} (\mathbf{P1}) \quad \min_{w_i} \quad & \sup_{r_k} f = \sum_{i=1}^{p} \int_{\Omega_i} (\overline{U} - y_i(w_i, r_k; t))dt && (23) \\ s.t. \quad & 0 \leq w_i \leq \overline{w_i}, \quad 1 \leq i \leq M && (24) \\ & \sum_{i=1}^{M} w_i \leq \overline{W} && (25) \\ & \underline{r_k} \leq r_k \leq \overline{r_k}, \quad 1 \leq k \leq q && (26) \end{aligned}$$

where voltage y_i is a function of w_i, r_k, and time t and can be solved from (22) and (2).

Problem (**P1**) is a constrained min-max optimization problem. The *sup* operation over all random variables r_k is to find the worst-case noise violation measures for a given power network design. This operation guarantees that all P/G network designs satisfy the given design constrains while considering the temporal and logic-induced correlations as well as L_{eff} variation among ports. This is of particular use for ASIC-style designs, where the worst-case design performance has to be ensured for sign-off. The *min* operation over all decap sizes w_i is to find the optimal decap budgeting solution so that the worst-case noise violation is minimized.

IV. Algorithms

A. Iterative Alternative Programming with Guaranteed Convergence

Because there exists no general technique to solve the constrained min-max problem (**P1**) optimally, we resort to an effective iterative optimization strategy, which we call *iterative alternative programming* (IAP). That is, instead of solving the min-max problem (**P1**) directly, we solve it by iteratively solving the following two sub-problems alternatively.

The first sub-problem assumes that all decaps' sizes w_i are known, hence the worst-case noise can be obtained by solving the following optimization problem

$$\begin{aligned} (\mathbf{P2}) \quad \max_{r_k} \quad & f = \sum_{i=1}^{p} \int_{\Omega_i} (\overline{U} - y_i(w_i, r_k; t))dt && (27) \\ s.t. \quad & \underline{r_k} \leq r_k \leq \overline{r_k}, \quad 1 \leq k \leq q && (28) \end{aligned}$$

The second sub-problem assumes that all random variables r_k have fixed values, hence the decap sizes to achieve the minimum noise can be obtained by solving the following optimization problem

$$\begin{aligned} (\mathbf{P3}) \quad \min_{w_i} \quad & f = \sum_{i=1}^{p} \int_{\Omega_i} (\overline{U} - y_i(w_i, r_k; t))dt && (29) \\ s.t. \quad & 0 \leq w_i \leq \overline{w_i}, \quad 1 \leq i \leq M && (30) \\ & \sum_{i=1}^{M} w_i \leq W, && (31) \end{aligned}$$

where W is the total white space available. Problem (**P3**) is exactly the deterministic version of the original problem formulation (**P1**).

We illustrate our idea in Figure 3 and the overall algorithm can be described in Algorithm 1, where *iter* is the current iteration number and ϵ determines the stop criteria of the optimization procedure. For each iteration, we increase the available white space by ΔW if the stopping criteria has not been met yet, which means the current white space is not enough.

The algorithm terminates when the change of objective function $|\Delta f|$ is sufficiently small, or we have used up all the white space. The first case corresponding to the situation where we have reduced noise below the bound before all the white space are used up, while the second case indicates that we have used up all the white space.

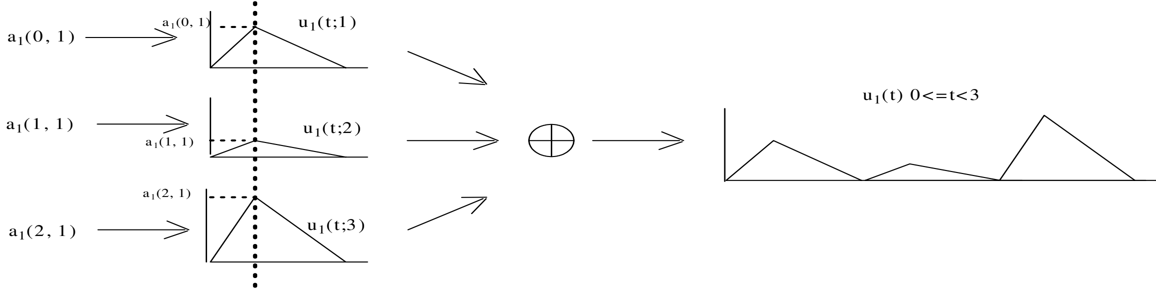

Fig. 2. An example to construct $u_1(t)$ for $L = 3$, $T = 1$ and $p = 1$ by aligning independent current components according to the time sequence.

In either case, the algorithm will terminates and the convergence of our algorithm is guaranteed. As shown in Figure 4, the choice of ΔW reflects a tradeoff between the runtime and the solution quality. Smaller ΔW can result in smaller noise under the same decap area but the runtime is increased as well. Setting $\Delta W = 0.004W$ gives a good balance in our experiment.

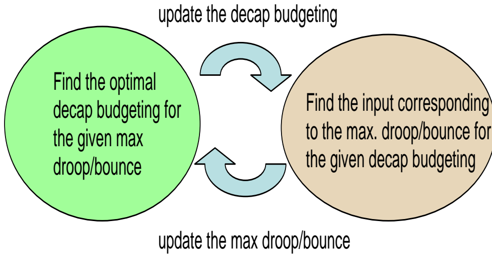

Fig. 3. Solve the min-max problem by iteratively solve two sub-problems.

Algorithm 1 Iterative alternative programming.

INPUT: initial w_i, r_k, current white space $\bar{W}$;
OUTPUT: final solution w_i to problem (**P1**);
Initialize: The current white space available $W = 0$;
for $iter = 0$; $|\Delta f| \leq \epsilon$ and W $\leq \bar{W}$; $iter++$ **do**
  $W = W + \Delta W$;
  w_i = solve-P3($iter$, w_i, r_k, W);
  r_k = solve-P2($iter$, w_i, r_k, W);
  Compute objective function with new r_k and w_i;
end for

B. Efficient Sequential Programming

Both problems (**P2**) or (**P3**) are constrained nonlinear optimization problems, and there exits no general technique to solve them efficiently. Because the constraints in both problems are linear, if we can approximate the objective function f by a first-order linear function, the original problems would become linear programming (LP) problems[3]. Because efficient solvers exist for LP problems, we can solve the approximated problems more efficiently than solving the original problems directly. Therefore, we propose to solve the original (**P2**) or (**P3**) problem via sequential linear programming (sLP).

For now, let us assume that we know how to compute the first-order sensitivities of the objective function f with respect to changing

[3]We can also extend our technique to approximate the objective function f by a second-order quadratic function, then the problem would become a quadratic programming (QP) problem. Our initial experimental results show that compared with LP, QP can further reduce the noise by 19% with the same decap area at the cost of $3X$ increased runtime. This will be reported in detail in the future.

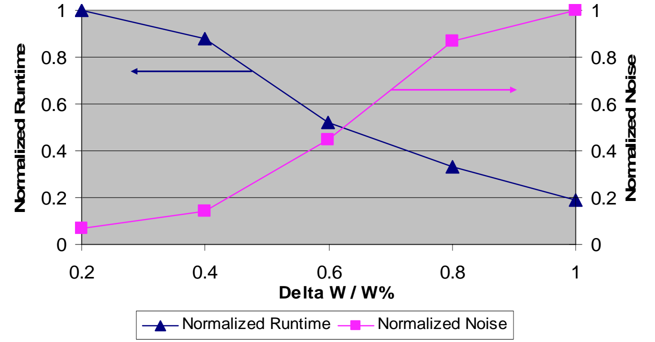

Fig. 4. The normalized runtime and noise w.r.t different $\frac{\Delta W}{W}$.

variables, which will be discussed in Section 4.3. Therefore, we can easily obtain the linear approximation of the objective function. For example, for the objective function in problem (**P3**), the changing variables are all Δw_i. Therefore, we have the following linear approximation for the objective function

$$f \quad \approx \quad f_0 + \sum_{i=1}^{M} \frac{\partial f}{\partial w_i} \Delta w_i, \tag{32}$$

where f_0 is the current value of the objective function, and $\frac{\partial f}{\partial w_i}$ are the first-order sensitivities of f. Apparently, (32) is a linear function of Δw_i. By replacing (27) with (32), we obtain an approximated LP formulation for (**P3**).

A high-level description of the sequential programming algorithm to solve either problem (**P2**) or (**P3**) is shown in Algorithm 2, where *iter2* is the current iteration number, *ITER2* is the maximum iteration bound. The iterations stop when the change of objective function $|\Delta f|$ is smaller than ϵ_2, which is dynamically adjusted according to the iteration number *iter* in the outer-loop of Algorithm 1. We employ an exponential decreasing function to adjust ϵ_2 in this work. The idea is that when the out-loop iteration is small (or we are far from the optimal solution), we can have an early termination of the inner-loop optimization procedure as shown in Algorithm 2 early. But when the outer-loop iteration becomes large enough (or we are close to the optimal solution), we should spend more time in each inner-loop optimization to find a better global optimal solution. Parameter η is used to control the efforts that we should spend in the inner-loop's optimization.

C. Sensitivity Computation

To solve (**P2**) and (**P3**) via sLP, we need to compute the sensitivities of the objective function f with respect to the design variables, i.e.,

Algorithm 2 Sequential linear programming for solving (**P2**) and (**P3**).

INPUT: $iter$, w_i, r_i, W;
OUTPUT: updated w_i for (**P3**) or r_i for (**P2**);
ϵ_2 = exp(-η·iter);
for $iter2$=0; $|\Delta f| \leq \epsilon_2$ or $iter2 \leq ITER2$; $iter2$++ **do**
  Compute the first-order sensitivities of f;
  Formulate (**P2**) or (**P3**) as an LP problem;
  Call LP solver to solve the above problem;
  Compute objective function with new w_i (**P2**) or r_i (**P3**);
end for

either w_i or r_i. Because this computation is similar for both (**P2**) and (**P3**), we will focus our discussion on (**P3**) in the following.

The first-order sensitivities of the objective function f of problem (**P3**) are defined as

$$\frac{\partial f}{\partial w_i} = -\sum_{i=1}^{p} \int_{\Omega_i} \frac{\partial y_i}{\partial w_i} dt = -\sum_{i=1}^{p} \int_{\Omega_i} L_{0i}^T \frac{\partial x}{\partial w_i} dt, \tag{33}$$

For simplicity of presentation, we have loosely applied the derivative notation on a vector for component-wise derivative. To compute the sensitivity of f w.r.t. w_i, all we need to know is the sensitivity of the state vector x with respect to w_i. We use Taylor expansion to express x as follows

$$x = x_0 + \sum_{i=1}^{M} \alpha_i \Delta w_i + \ldots, \tag{34}$$

where α_i is the first-order sensitivity of x w.r.t. random variable w_i, i.e., we have

$$\frac{\partial x}{\partial w_i} = \alpha_i. \tag{35}$$

To compute these sensitivities, we recognize that x also satisfies the differential equation given by the parameterized MNA formulation (22). By Laplace transformation, we re-write (22) as follows

$$(G + \sum_{i=1}^{M} \Delta w_i \cdot G_{w,i})x + s(C + \sum_{i=1}^{M} \Delta w_i \cdot C_{w,i})x = Bu. \tag{36}$$

By plugging (34) into (36), we obtain terms of Δw_i with different orders. By equating the zero-order terms of Δw_i from both left and right hand sides in (36), we obtain a set of equations as follows

$$(G + sC)x_0 = Bu. \tag{37}$$

By equating the first-order terms of Δw_i, we obtain sets of equations as follows for all $1 \leq i \leq M$

$$(G + sC)\alpha_i = -(G_{w,i} + sC_{w,i})x_0. \tag{38}$$

By applying the Backward Euler integration formula and assuming the time step as h, we can re-write (37) and (38) as follows

$$(G + \frac{C}{h})x_0(t+h) = Bu(t+h) + x_0(t)\frac{C}{h}, \tag{39}$$

$$(G + \frac{C}{h})\alpha_i(t+h) = -(G_{w,i} + \frac{C_{w,i}}{h})x_0(t+h) + \frac{x_0(t)C_{w,i} + \alpha_i(t)C}{h}. \tag{40}$$

Because all equations in (39) and (40) share the same left-hand side matrix, $(G + C/h)$, we only need to perform LU-factorization once, and then reuse the same factorization to solve for x_0 and α_i sequentially at each time step. This computation is efficient because it only involves some matrix-vector multiplications, and backward and forward substitutions.

The integral interval Ω_i for port i is decided by x_0. Once x_0 is solved, we have $y = L_0^T x_0$, and then the corresponding interval can be decided from (4). By doing so we have assumed that the incremental δw_i is relatively small in each step and will not significantly influence the integral interval. In summary, we can compute the first-order sensitivities of the objective function f of problem (**P3**) by following the Algorithm 3.

Algorithm 3 Sensitivity computation for (**P3**).

INPUT: w_i, r_k, h, T;
OUTPUT: f and α_i;
factorization: LU factorize $G + C/h$;
for $t = 0$; $t + h \leq T$; $t = t + h$ **do**
  Solve (39) for $x_0(t+h)$;
end for
for $i = 1$; $i \leq p$; $i++$ **do**
  Use (4) to compute Ω_i from $y(t) = L_0^T x_0(t)$;
end for
for $t = 0$; $t + h \leq T$; $t = t + h$ **do**
  Solve (40) for $\alpha_i(t+h)$;
  Solve $\frac{\partial f}{\partial w_i}$ from (33);
end for

V. Experimental Results

In this section, we present experiments using four industrial P/G network designs. For each benchmark, we randomly select 20% of total nodes as candidate nodes for decap insertion, i.e., $M = 20\%N$. For fair comparison, when comparing the runtime and noise, the same white space is used up for different methods. We run experiments on a LINUX workstation with Pentium IV 2.66G CPU and 1G RAM. We partition the circuits according to the method in [15]. We use the package FASTICA [20] to perform ICA. Finally, we use MOSEK as the linear/quadratic programming solver [21] and random walk based simulator [22] with detailed (not triangular) input current waveform to obtain the noise reported in this section.

A. Decap Budgeting without Leff variation

We compare three current models as shown in Table II: maximum currents at all ports (model 1), stochastic model (model 2) with logic-induced correlation only ($L = 1$), and stochastic model (model 3) with both logic-induced and temporal correlation. For temporal correlation, we always use $L = 4$ since all circuits tested take at most four clock cycles to complete any one instruction. Table II reports the noise and runtime for the four benchmarks with different number of nodes. Compared with the baseline model with maximum currents at all ports [4], the model considering temporal correlation reduces noise by up to 5×; and the model considering both temporal and logic-induced correlations reduces noise by up to 17× (see bold in Table II). This is because the first two models cannot model the currents effectively and lead to inserting unnecessarily large decaps in some regions. Thus, they result in more noise in the other regions since the total decap area is given. As for the runtime, model 2 needs about 1.5× more time than model 1, while model 3 needs about 2.3× more. The runtime overhead is the price we have to pay in order to achieve better designs.

In Fig. 5, we plot the time-domain responses at one randomly selected port for two optimization iterations by alternatively solving the problem (**P3**) and (**P2**). The benchmark has 1284 nodes. The initial waveform is denoted by "A0:initial". After performing decap sizing once by solving problem (**P3**) with a fixed choice of random variables r_k, we obtain the new waveform as denoted by "A1:(**P3**)".

[4] We solve it by iteratively solving (**P3**) without altering to (**P2**).

TABLE II

NOISE AND RUNTIME COMPARISON BETWEEN THE THREE MODELS.

	Model 1	maximum currents at all ports					
	Model 2	stochastic model with logic-induced correlation					
	Model 3	Model 2 + temporal correlation					
Node #	Port #	noise (V*s)			runtime (s)		
		model 1	model 2	model 3	model 1	model 2	model 3
1284	**426**	**6.33e-7**	**1.28e-7**	**4.10e-8**	**104.2**	**161.2**	**282.3**
10490	3398	5.21e-5	1.09e-5	4.80e-6	973.2	1430	2199
42280	13327	7.92e-4	5.38e-4	9.13e-5	2732	3823	5238
166380	42146	1.34e-2	5.37e-3	2.28e-3	3625	5798	7821
avg		1	$1/3\times$	$1/9\times$	1	$1.50\times$	$2.26\times$

We then switch to solve problem (**P2**) by varying the values of those random variables r_k, but with fixed decap sizes w_i. We see that the waveform of the final worst-case voltage drop becomes worse compared to the deterministic solution; hence we obtain a new voltage drop waveform as denoted by "A2:(**P2**)". We then switch back to solve the decap sizing problem (**P3**) with fixed but newly updated choice of random variables r_k. At the end of this optimization, we arrive at a new voltage waveform as denoted by "A3:(**P3**)". Apparently, compared to "A1:(**P3**)", the new solution has smaller voltage drop. If we continue the same procedures by following the IAP algorithm given in Fig. 1, similar sequences of time domain voltage drop waveforms would repeat as we have shown in Fig. (5) until we converge to an optimal solution. Also, the voltage drop is reduced mostly in the first optimization iteration denoted as "A1:(**P3**)". Afterward, the voltage drop reduction is relatively small. This observation is in agreement with the common knowledge about any sensitivity-based optimization techniques. In this particular example, we find that the first two iterations reduces the noise by 51.4%.

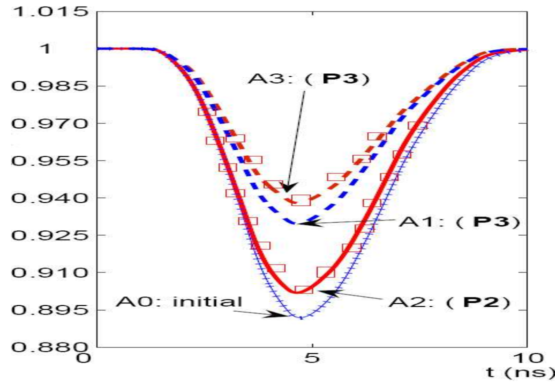

Fig. 5. Time domain waveforms at one port after sLP for different iterations.

B. L_{eff} Variation Aware Decap Budgeting

In the presence of process variation, we want to minimize the worst-case noise for L_{eff} variation. We solve this via the proposed IAP technique in Algorithm 1. We denote our L_{eff} variation aware approach as $sLP + L_{eff}$ and the counterpart as sLP. Before we quantitatively compare the two methods, we first use Figure 6 and Figure 7 to demonstrate the effectiveness of L_{eff} variation aware decap budgeting. In Figure 6 we use the sLP approach and design for the deterministic case without process variation. We plot the noise map over different ports. The same circuit is used in plot (a) and plot (b), but plot (a) has no L_{eff} variation while plot (b) has 15% L_{eff} variation. As we can see, the noise map changes significantly from (a) to (b). The noise of (a) is $1.44 \times 10^{-4}\ V * s$, while that of (b) is $5.9 \times 10^{-4}\ V * s$ (4× increase). which may lead to noise violation in (b). In contrast, Figure 7 uses the same circuit optimized by the $sLP + L_{eff}$ approach. For the noise maps, plot (a) considers no variation, and plot (b) has 15% L_{eff} variation. We can see that the two noise maps have little difference. The noise of (a) is 1.17×10^{-4} $V * s$, and that of (b) is $1.24 \times 10^{-4}\ V * s$ (1.06× increase), which means that the design in Figure 7 is robust. Comparing Figure 6 and Figure 7, one can clearly see that the noise in Figure 7 is much smaller than that in Figure 6, although both have the same decap area constraints.

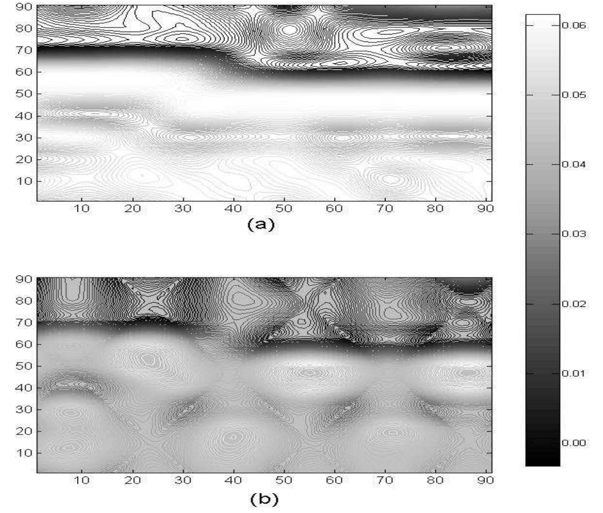

Fig. 6. The noise maps for the an industry power mesh with decap budgeting using sLP. The circuits in (a) and (b) are the same but (b) considers 15% L_{eff} variation.

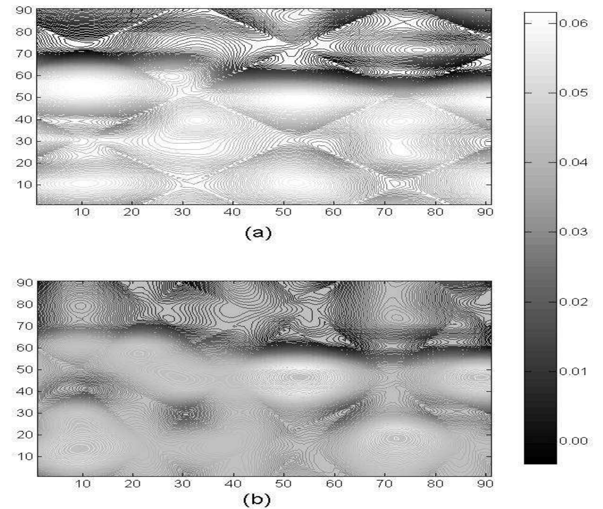

Fig. 7. The noise maps for the same power mesh with decap budgeting using $sLP + L_{eff}$. The circuits in (a) and (b) are the same but (b) considers 15% L_{eff} variation.

Next we compare the mean value μ and 3σ value of the noise distribution with 10% L_{eff} variation based on Monte Carlo simulation with 10,000 runs, and the results are reported in Table III. Compared

TABLE III

THE MEAN VALUE μ, 3σ VARIANCE OF THE NOISE AND RUNTIME (RT) COMPARISON BETWEEN $sLP + L_{eff}$ AND sLP WITH 10% INTRA-DIE L_{eff} VARIATION.

Node #	Port #	sLP			$sLP + L_{eff}$		
		μ (V*s)	3σ (V*s)	RT (s)	μ (V*s)	3σ (V*s)	RT (s)
1284	426	9.28e-7	3.97e-7	184.2	6.14e-7	1.38e-7	332.8 ($1.81\times$)
10490	3398	1.03e-4	4.79e-5	1121	7.22e-5	1.23e-5	3429 ($3.06\times$)
42280	13327	2.29e-3	9.72e-4	2236	8.23e-4	1.01e-4	6924 ($3.10\times$)
166380	**42146**	**2.06e-2**	**9.91e-3**	**3824**	**5.31e-3**	**8.32e-4**	**11224** ($2.93\times$)
avg		1	1	1	$1/2\times$	$1/5\times$	$2.73\times$

with using deterministic L_{eff}, considering L_{eff} variation reduces the mean noise by up to $4\times$ and 3σ noise by up to $13\times$ (see bold in Table III), when both applying the current model with temporal and logic-induced correlations. As for the runtime between sLP and $sLP + L_{eff}$, the latter needs about $2.7\times$ more time than the former on average.

VI. CONCLUSIONS AND FUTURE WORK

As an example of optimization for both operation and process variations, this paper has solved the on-chip decoupling capacitance (decap) budgeting problem to minimize time-domain power noise integral (in short, noise) subject to a given decap area constraint. We have developed a novel stochastic current model to efficiently and accurately capture operation variation such as temporal correlation between clock cycles, and logic-induced correlation between ports. The model also considers process (L_{eff}) variation with spatial correlation. We have also proposed an iterative alternative programming algorithm to solve the decap budgeting problem under the stochastic current model, and conducted experiments using industrial examples. We have shown that compared with the baseline model assuming maximum currents at all ports and under the same decap area constraint, the model considering temporal correlation reduces the noise by up to $5\times$, and the model considering both temporal and logic-induced correlations reduces the noise by up to $17\times$. Compared with the model using deterministic process parameters, considering L_{eff} variation reduces the mean noise by up to $4\times$ and the 3σ noise by up to $13\times$.

Different from the existing stochastic optimization used mainly for process variation, this paper to the best of our knowledge is the first in-depth study on stochastic optimization taking into account both operation and process variations for power network design. This paper convincingly demonstrates that stochastic optimization considering operation variation is highly beneficial for power integrity. Because operation variation affects signal and power integrity, this paper opens a new direction for future research to optimize signal and thermal integrity.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their useful suggestion and comments to help improve the quality of this work. This paper is partially supported by NSF CAREER award and a UC MICRO grant sponsored by Altera, RIO and Intel. Address comments to lhe@ee.ucla.edu.

REFERENCES

[1] K. Agarwal and S. Nassif, "Characterizing Process Variation in Nanometer CMOS," in *IEEE/ACM DAC*, 2007.

[2] M. Mani, A. Devgan, and M. Orshansky, "An Efficient Algorithm for Statistical Minimization of Total Power under Timing Yield Constraints," in *IEEE/ACM DAC*, 2005.

[3] S. Bhardwaj and S. B. K. Vrudhula, "Leakage Minimization of Nanoscale Circuits in the Presence of Systematic and Random Variations," in *IEEE/ACM DAC*, 2005.

[4] D. Sinha, N. V. Shenoy, and H. Zhou, "Statistical Gate Sizing for Timing Yield Optimization," in *IEEE/ACM ICCAD*, 2005.

[5] A. Davoodi and A. Srivastava, "Variability-Driven Gate Sizin for Binning Yield Optimization," in *IEEE/ACM DAC*, 2006.

[6] L. He, A. Kahng, K. H. Tam, and J. Xiong, "Simultaneous Buffer Insertion and Wire Sizing Considering Systematic CMP Variation and Random Leff Variation," *IEEE Trans. on CAD*, 2007.

[7] M. Mani, A. Singh, and M. Orshansky, "Joint Design-Time and Post-Silicon Minimization of Parametric Yield Loss using Adjustable Robust Optimization," in *IEEE/ACM ICCAD*, 2006.

[8] M. Zhao, R. Panda, S. Sundareswaran, S. Yan, and Y. Fu, "A fast on-chip decoupling capacitance budgeting algorithm using macromodeling and linear programming," in *IEEE/ACM DAC*, 2006.

[9] H. Su, S. S. Sapatnekar, and S. R. Nassif, "Optimal decoupling capacitor sizing and placement for standard-cell layout designs," *IEEE Trans. on CAD*, vol. 22, pp. 428–436, April 2003.

[10] J. Fu, Z. Luo, X. Hong, Y. Cai, S.-D. Tan, and Z. Pan, "A fast decoupling capacitor budgeting algorithm for robust on-chip power delivery," in *Proc. Asia South Pacific Design Automation Conf. (ASPDAC)*, pp. 505–510, Jan. 2004.

[11] H. Li, Z. Qi, S. X.-D. Tan, L. Wu, Y. Cai, and X. Hong, "Partitioning-based approach to fast on-chip decap budgeting and minimization," in *IEEE/ACM DAC*, pp. 170–175, June 2005.

[12] S. Pant, D.Blaauw, V. Zolotov, S.Sundareswaran, and R. Panda, "A stochastic approach to power grid analysis," in *IEEE/ACM DAC*, 2004.

[13] I. A. Ferzli and F. N. Najm, "Statistical verification of power grids considering process-induced leakage current variations," in *IEEE/ACM ICCAD*, 2003.

[14] C. Visweswariah, R. A. Haring, and A. R. Conn, "Noise Consierations in Circuit Optimization," *IEEE Trans. on CAD*, 2000.

[15] D. Kouroussis, I. A. Ferzli, and F. N. Najm, "Incremental partitioning-based vectorless power grid verification," in *IEEE/ACM ICCAD*, 2005.

[16] M. Orshansky, L. Milor, P. Chen, K. Keutzer, and C. Hu, "Impact of Spatial Intrachip Gate Length Variability on the Performance of High-speed Digital Circuits," *IEEE Trans. on CAD*, 2002.

[17] Y. Cao and L. T. Clark, "Mapping statisitical process variations toward circuit performance variability: An analytical modeling approach," in *IEEE/ACM DAC*, 2005.

[18] A. Hyvarinen, J. Karhunen, and E. Oja, *Independent Component Analysis*. John Wiley & Sons, 2001.

[19] H. Zheng, B. Krauter, and L. Pileggi, "On-Package Decoupling Optimization with Package Macromodels," in *Proc. IEEE Custom Integrated Circuits Conference (CICC)*, 2003.

[20] A. Hyvarinen and E. Oja, "A Fast Fixed-Point Algorithm for Independent Component Analysis," *Neural Computation*, 1997.

[21] http://www.mosek.com

[22] H. Qian, S. R. Nassif, and S. S. Sapatnekar, "Power Grid Analysis Using Random Walks," *IEEE Trans. on CAD*, 2005.