Efficient SRAM Failure Rate Prediction via Gibbs Sampling

Shupeng Sun, *Student Member, IEEE,* Yamei Feng, Changdao Dong, and Xin Li, *Senior Member, IEEE*

***Abstract*—Statistical analysis of SRAM has emerged as a challenging issue because the failure rate of SRAM cells is extremely small. In this paper, we develop an efficient importance sampling algorithm to capture the rare failure event of SRAM cells. In particular, we adapt the Gibbs sampling technique from the statistics community to find the optimal probability distribution for importance sampling with a low computational cost (i.e., a small number of transistor-level simulations). The proposed Gibbs sampling method applies an integrated optimization engine to adaptively explore the failure region in a Cartesian or spherical coordinate system by sampling a sequence of 1-D probability distributions. Several implementation issues such as 1-D random sampling and starting point selection are carefully studied to make the Gibbs sampling method efficient and accurate for SRAM failure rate prediction. Our experimental results of a 90 nm SRAM cell demonstrate that the proposed Gibbs sampling method achieves 1.4–4.9× runtime speedup over other state-of-the-art techniques when a high prediction accuracy is required (e.g., the relative error defined by the 99% confidence interval reaches 5%). In addition, we further demonstrate an important example for which the proposed Gibbs sampling algorithm accurately estimates the correct failure probability, while the traditional techniques fail to work.**

***Index Terms*—Failure rate, Gibbs sampling, Monte Carlo analysis, process variation, SRAM.**

I. Introduction

As DEEP SUBMICRON technology advances, process variations pose a new set of challenges on SRAM design. SRAM has been widely embedded in a large amount of semiconductor chips. For example, roughly half of the area of an advanced microprocessor chip is occupied by SRAM [19]. SRAM cells are generally designed with minimum-size devices [19] and can be significantly impacted by large-scale process variations (e.g., local mismatches caused by random doping fluctuations) at nanoscale technology [2]–[4]. For this reason, it becomes increasingly critical to evaluate the failure rate of SRAM cells both efficiently and accurately in order to achieve a robust design.

Toward this goal, a number of statistical analysis methods have been proposed for SRAM circuits [5]–[15]. For instance, analytical performance models have been derived to predict SRAM parametric yields [5]–[7]. While these models offer great design insights to understand SRAM circuits, they may not accurately capture the circuit behavior due to various approximations that are made. Another possible approach for SRAM failure rate prediction is based on transistor-level simulation, including both Monte Carlo analysis [8]–[14] and deterministic failure region prediction [15].

Since SRAM cells typically have extremely small failure probability (e.g., $10^{-8} \sim 10^{-6}$), a simple Monte Carlo method by directly sampling the variation space suffers from slow convergence, as only a few random samples will fall into the failure region. To improve the sampling efficiency, a number of importance sampling methods have been proposed for fast SRAM failure rate prediction [8]–[14]. The key idea of importance sampling is to directly sample the failure region based on a distorted probability density function (PDF) instead of the original PDF of process variations.

Applying importance sampling to SRAM analysis, however, is not trivial. The efficiency of importance sampling heavily relies on the choice of the distorted PDF that is used to generate random samples. Ideally, in order to maximize prediction accuracy, we should sample the failure region that is most likely to occur. Such a goal, however, is extremely difficult to achieve, since we never exactly know the failure region in practice. The challenging issue here is how to determine the optimal PDF for importance sampling so that the SRAM failure rate can be efficiently predicted.

In this paper, a novel Gibbs sampling method is proposed to improve the efficiency of SRAM failure rate prediction. Unlike the traditional Monte Carlo algorithm that samples a given PDF, the proposed Gibbs sampling approach does not need to know the sampling PDF explicitly. Instead, it adaptively searches the failure region and then generates random samples in it. When applied to SRAM failure rate analysis, Gibbs sampling can be conceptually viewed as a unique Monte Carlo method with an integrated optimization engine which allows us to efficiently explore the failure region. As a result, SRAM failure probability can be accurately predicted with a small number of sampling points (i.e., a small number of transistor-level simulations). Our experimental results of a 90 nm SRAM cell demonstrate that compared to other

Manuscript received January 3, 2012; revised April 29, 2012 and June 28, 2012; accepted July 3, 2012. Date of current version November 21, 2012. This work was supported in part by Semiconductor Research Corporation, under Contract 1836.044, and by the National Science Foundation, under Contract CCF-1016890. This paper was presented in part at the Design Automation Conference in 2011 [1]. This paper was recommended by Associate Editor S. Vrudhula.

The authors are with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213 USA (e-mail: shupengs@ece.cmu.edu; yameif@ece.cmu.edu; changdao@ece.cmu.edu; xinli@ece.cmu.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCAD.2012.2209884

state-of-the-art techniques, the proposed Gibbs sampling algorithm achieves 1.4–4.9× runtime speedup when a high prediction accuracy is required (e.g., the relative error defined by the 99% confidence interval reaches 5%). In addition, we further demonstrate an important example for which the proposed Gibbs sampling algorithm accurately estimates the correct failure probability, while the traditional techniques fail to work.

While the Gibbs sampling method was initially developed by the statistics community [16], [22], it is particularly tuned for our SRAM analysis applications via four important new contributions. First, the proposed Gibbs sampling algorithm is implemented for both Cartesian and spherical coordinate systems to optimally explore the failure region. The accuracy achieved by these two different implementations is application-dependent, as will be demonstrated by the experimental results in Section V. A novel variable mapping scheme is derived to facilitate efficient statistical sampling in spherical coordinate systems. In particular, we define $M + 1$ random variables to specify the spatial location in an M-dimensional spherical coordinate system. Each of these $M + 1$ random variables follows a well-known statistical distribution and can be efficiently sampled with a low computational cost.

Second, as previously mentioned, Gibbs sampling iteratively searches the failure region. At each iteration step, it needs to generate a random sample from an irregular 1-D PDF that is not simply uniform or Normal. Such a sampling task cannot be easily done by directly using a random number generator. For this reason, we propose to adopt the inverse-transform method [22] and incorporate it into our proposed Gibbs sampling engine. As such, randomly sampling an arbitrary 1-D PDF can be performed efficiently.

Third, an efficient implementation of Gibbs sampling requires a good starting point to speed up the convergence. This is similar to most optimization algorithms where a good starting point facilitates fast convergence. We propose a model-based optimization to determine a good starting point so that the Gibbs sampling algorithm can converge quickly (i.e., accurately predict the failure probability with few sampling points).

Finally, to further improve prediction accuracy and reduce computational cost, a two-stage Monte Carlo flow is developed where Gibbs sampling is applied to create a set of random samples during the first stage, and these samples are used to "learn" the joint PDF for importance sampling. Next, during the second stage, a large number of random samples are directly generated from the PDF learned from the first stage to accurately estimate the failure probability.

The remainder of this paper is organized as follows. In Section II, we briefly review the background on importance sampling, and then propose the Gibbs sampling method in Section III. Several implementation issues are discussed in detail in Section IV. A 90 nm SRAM cell is used to demonstrate the efficacy of the proposed Gibbs sampling method in Section V. Finally, we conclude in Section VI.

II. Background

Suppose that $\mathbf{x} = [x_1\, x_2 \ldots x_M]^T$ is an M-dimensional random variable modeling process variations and its joint PDF is $f(\mathbf{x})$. Typically, $\mathbf{x}$ is modeled as a multivariate Normal distribution [5]–[15]. Without loss of generality, we further assume that the random variables $\{x_m; m = 1, 2, \ldots, M\}$ in the vector $\mathbf{x}$ are mutually independent and standard Normal (i.e., with zero mean and unit variance)

$$f(\mathbf{x}) = \prod_{m=1}^{M} \left[\frac{1}{\sqrt{2\pi}} \cdot \exp\left(-\frac{x_m^2}{2} \right) \right]. \tag{1}$$

Any correlated random variables that are jointly Normal can be transformed to the independent random variables $\{x_m; m = 1, 2, \ldots, M\}$ by principal component analysis [20].

The failure probability of an SRAM cell can be mathematically represented as follows [8]:

$$P_f = \int_{\Omega} f(\mathbf{x}) \cdot d\mathbf{x} \tag{2}$$

where Ω denotes the failure region, i.e., the subset of the variation space where the performance of interest (e.g., read margin, write margin) does not meet the specification. Alternatively, the failure probability in (2) can be defined as follows:

$$P_f = \int_{-\infty}^{+\infty} I(\mathbf{x}) \cdot f(\mathbf{x}) \cdot d\mathbf{x} \tag{3}$$

where $I(\mathbf{x})$ represents the indicator function

$$I(x) = \begin{cases} 1 & \mathbf{x} \in \Omega \\ 0 & \mathbf{x} \notin \Omega. \end{cases} \tag{4}$$

The failure probability P_f can be estimated by Monte Carlo analysis. The key idea is to draw N random samples from $f(\mathbf{x})$, and then compute the mean of these samples

$$\tilde{P}_f^{MC} = \frac{1}{N} \cdot \sum_{n=1}^{N} I\left[\mathbf{x}^{(n)}\right] \tag{5}$$

where $\mathbf{x}^{(n)}$ is the nth random sample generated by Monte Carlo analysis.

For our proposed SRAM application, the failure probability P_f in (3) is extremely small (e.g., 10^{-8}–10^{-6}) and most random samples created by Monte Carlo analysis do not fall into the failure region Ω. Hence, a large number of (e.g., over 10^7–10^9) samples are needed by the Monte Carlo method to accurately estimate the failure rate. Note that an expensive transistor-level simulation is required to create each sampling point. In other words, 10^7–10^9 simulation runs must be performed in order to collect 10^7–10^9 random samples. It, in turn, implies that the aforementioned Monte Carlo method is extremely expensive, when applied to most SRAM analysis problems.

To address this computational cost issue, importance sampling has been proposed to improve the efficiency of Monte Carlo analysis [8]–[14]. It aims to directly generate a large number of random samples in the failure region by using a distorted PDF $g(\mathbf{x})$. In this case, the failure probability can be expressed as follows [8]:

$$P_f = \int_{-\infty}^{+\infty} \frac{I(\mathbf{x}) \cdot f(\mathbf{x})}{g(\mathbf{x})} \cdot g(\mathbf{x}) \cdot d\mathbf{x}. \tag{6}$$

In other words, (6) calculates the expected value of the function $I(\mathbf{x})\cdot f(\mathbf{x})/g(\mathbf{x})$ where the random variable $\mathbf{x}$ follows the PDF $g(\mathbf{x})$. If N sampling points $\{\mathbf{x}^{(n)}; n = 1, 2, \ldots, N\}$ are drawn from $g(\mathbf{x})$, the failure probability in (6) can be estimated by [8]

$$\tilde{P}_f^{IS} = \frac{1}{N} \cdot \sum_{n=1}^{N} \frac{I\left[\mathbf{x}^{(n)}\right] \cdot f\left[\mathbf{x}^{(n)}\right]}{g\left[\mathbf{x}^{(n)}\right]}. \quad (7)$$

Note that the estimated failure probabilities in (5) and (7) are identical, if and only if the number of random samples (i.e., N) is infinite. In practice, when a finite number of sampling points are available, the results from (5) and (7) can be substantially different. If the distorted PDF $g(\mathbf{x})$ is properly chosen for importance sampling, (7) can be much more accurate than the simple Monte Carlo method in (5). In theory, the optimal PDF $g(\mathbf{x})$ leading to maximum estimation accuracy is [22]

$$g^{OPT}(\mathbf{x}) = \frac{I(\mathbf{x}) \cdot f(\mathbf{x})}{P_f}. \quad (8)$$

Intuitively, if the PDF $g^{OPT}(\mathbf{x})$ in (8) is used, the function $I(\mathbf{x})\cdot f(\mathbf{x})/g^{OPT}(\mathbf{x})$ becomes a constant with zero variance. Hence, its expected value can be accurately estimated by (7) using few random samples.

Studying (8) reveals two important properties of the optimal PDF $g^{OPT}(\mathbf{x})$. First, $g^{OPT}(\mathbf{x})$ is nonzero if and only if the variable $\mathbf{x}$ sits in the failure region. It, in turn, implies that we should directly sample the failure region to achieve maximum accuracy. Second, $g^{OPT}(\mathbf{x})$ is proportional to the original PDF $f(\mathbf{x})$ of process variations. Namely, the entire failure region should *not* be sampled uniformly. Instead, we should sample the variation space where failure is most likely to occur.

In practice, however, sampling the optimal PDF $g^{OPT}(\mathbf{x})$ in (8) is not trivial, as the indicator function $I(\mathbf{x})$ is unknown. Most existing importance sampling algorithms apply various heuristics to approximate the optimal PDF $g^{OPT}(\mathbf{x})$ [8]–[14]. In this paper, we propose a new Gibbs sampling method that adaptively samples the optimal PDF $g^{OPT}(\mathbf{x})$ without explicitly knowing the indicator function $I(\mathbf{x})$. As such, the SRAM failure rate can be accurately predicted with a low computational cost.

III. Gibbs Sampling

As described in Section II, directly sampling the optimal PDF $g^{OPT}(\mathbf{x})$ in (8) is difficult for two reasons. First, the indicator function $I(\mathbf{x})$ is not known in advance, as the failure region is unknown. Second, since $g^{OPT}(\mathbf{x})$ is not a simple multivariate statistical distribution such as uniform distribution or Normal distribution, it is extremely difficult, if not impossible, to directly draw random samples from $g^{OPT}(\mathbf{x})$.

In this paper, we adopt the Gibbs sampling method [16], [22] from the statistics community to predict the failure probability of SRAM cells. Gibbs sampling provides two promising features, compared to other traditional techniques [8]–[14]. First, it can efficiently search the failure region and determine the indicator function $I(\mathbf{x})$ on the fly. From this point of view, Gibbs sampling can be conceptually viewed as an integrated optimization engine that allows us to adaptively sample the optimal PDF $g^{OPT}(\mathbf{x})$ in (8).

Second, Gibbs sampling does not directly draw random samples from a multidimensional joint PDF. Instead, it iteratively samples a sequence of 1-D PDFs. These 1-D PDFs are not simply uniform or Normal and, hence, cannot be directly sampled by a simple random number generator. However, as will be demonstrated in Section IV-A, the aforementioned 1-D sampling can be efficiently implemented with an inverse-transform method [22] with a low computational cost.

Our proposed Gibbs sampling method can be implemented in both Cartesian and spherical coordinate systems. The accuracy achieved by these two different implementations is application-dependent. In what follows, we describe the technical details for both implementations.

A. Gibbs Sampling in a Cartesian Coordinate System

To intuitively illustrate the Gibbs sampling algorithm, we first consider the simple 2-D example in Fig. 1 where our goal is to sample the PDF $g^{OPT}(x_1, x_2)$. In this example, Gibbs sampling starts from an initial point $[x_1^{(1)}\ x_2^{(1)}]^T$. It first samples the conditional PDF $g^{OPT}[x_1|x_2^{(1)}]$ and replaces $x_1^{(1)}$ by a new value $x_1^{(2)}$, as shown in Fig. 1(a) and (b). During this iteration step, we generate a new sampling point $[x_1^{(2)}\ x_2^{(1)}]^T$. Next, Gibbs sampling samples a different conditional PDF $g^{OPT}[x_2|x_1^{(2)}]$ and replaces $x_2^{(1)}$ by a new value $x_2^{(2)}$, as shown in Fig. 1(c) and (d). A new sampling point $[x_1^{(2)}\ x_2^{(2)}]^T$ is created. Since the random variable $\mathbf{x}$ is 2-D in this example, Gibbs sampling varies x_1 again at the third iteration step. It draws a new random value $x_1^{(3)}$ by sampling the conditional PDF $g^{OPT}[x_1|x_2^{(2)}]$, resulting in a new sampling point $[x_1^{(3)}\ x_2^{(2)}]^T$. These iteration steps are repeatedly applied until a sufficient number of random samples are created. It can be proven that the aforementioned iteration yields a sequence of random samples that follow the given distribution $g^{OPT}(x_1, x_2)$ [16], [22].

The aforementioned 2-D Gibbs sampling can be extended to the general case where the PDF $g^{OPT}(\mathbf{x}) = g^{OPT}(x_1, x_2, \ldots, x_M)$ is M-dimensional. Starting from an initial point $[x_1^{(1)}\ x_2^{(1)} \ldots x_M^{(1)}]^T$, Gibbs sampling assigns a new value to one of the M random variables at each iteration step. This new value is determined by randomly sampling a conditional PDF. For instance, when sampling the mth random variable x_m, the following conditional PDF is used:

$$g^{OPT}(x_m|x_1, \cdots, x_{m-1}, x_{m+1}, \cdots, x_M) = g^{OPT}\left(x_m|\mathbf{x}_{\backslash m}\right) \quad (9)$$

where $\mathbf{x}_{\backslash m}$ denotes the vector $\mathbf{x}$ with x_m removed. Such random sampling is repeated with one random variable sampled at one time. Algorithm 1 summarizes the major steps of Gibbs sampling.

B. Gibbs Sampling in the Spherical Coordinate System

The Gibbs sampling method summarized by Algorithm 1 can be extended to spherical coordinate systems. Note that the accuracy achieved by the spherical coordinate implementation can be substantially different from that of the Cartesian coordinate implementation, depending on the application of

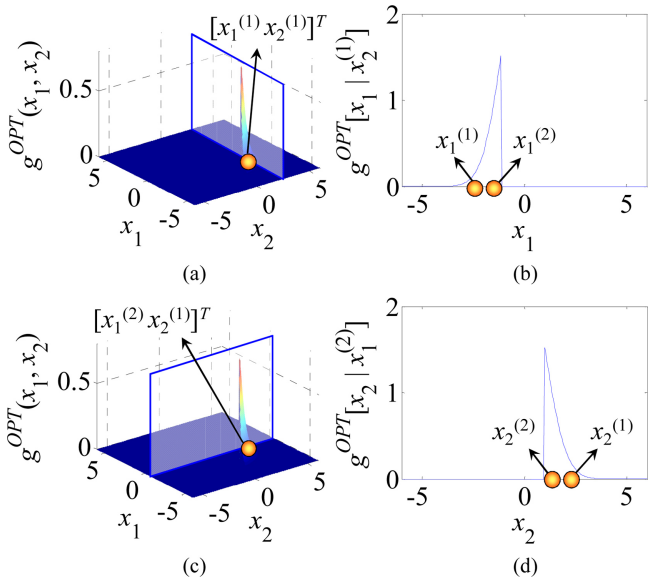


Fig. 1. Simple example of 2-D Gibbs sampling uses $[x_1^{(1)}\, x_2^{(1)}]^T$ as the starting point. (a) Intersection of the joint PDF $g^{OPT}(x_1, x_2)$ and the plane $x_2 = x_2^{(1)}$ defines the conditional PDF $g^{OPT}[x_1|x_2^{(1)}]$. (b) New sampling point $x_1^{(2)}$ is drawn from $g^{OPT}[x_1|x_2^{(1)}]$. (c) Intersection of the joint PDF $g^{OPT}(x_1, x_2)$ and the plane $x_1 = x_1^{(2)}$ defines the conditional PDF $g^{OPT}[x_2|x_1^{(2)}]$. (d) New sampling point $x_2^{(2)}$ is drawn from $g^{OPT}[x_2|x_1^{(2)}]$.

Algorithm 1 Gibbs sampling in Cartesian coordinate system

1. Start from an M-dimensional PDF $g^{OPT}(x_1, x_2, \ldots, x_M)$.
2. Select an initial starting point $[x_1^{(1)}\, x_2^{(1)} \ldots x_M^{(1)}]^T$.
3. For $t = 1, 2, \ldots$
4. For $m = 1, 2, \ldots, M$
5. Draw $x_m^{(t+1)}$ from the conditional PDF $g^{OPT}[x_m \mid x_1^{(t+1)}, \ldots, x_{m-1}^{(t+1)}, x_{m+1}^{(t)}, \ldots, x_M^{(t)}]$ to create a new sampling point $[x_1^{(t+1)} \ldots x_m^{(t+1)}\, x_{m+1}^{(t)} \ldots x_M^{(t)}]^T$.
6. End For
7. End For

interest. For instance, an SRAM example will be shown in Section V-B where the spherical coordinate implementation accurately estimates the failure probability while the Cartesian coordinate implementation fails to work.

Traditionally, M variables are used to specify the spatial location in an M-dimensional spherical coordinate system [21]

$$\begin{aligned} x_1 &= r \cdot \cos(\theta_1) \\ x_2 &= r \cdot \sin(\theta_1) \cdot \cos(\theta_2) \\ x_3 &= r \cdot \sin(\theta_1) \cdot \sin(\theta_2) \cdot \cos(\theta_3) \\ &\vdots \\ x_{M-1} &= r \cdot \sin(\theta_1) \cdot \cdots \cdot \sin(\theta_{M-2}) \cdot \cos(\theta_{M-1}) \\ x_M &= r \cdot \sin(\theta_1) \cdot \cdots \cdot \sin(\theta_{M-2}) \cdot \sin(\theta_{M-1}). \end{aligned} \tag{10}$$

Equation (10) shows the mapping from the spherical coordinate r and $\boldsymbol{\theta} = [\theta_1\, \theta_2 \ldots \theta_{M-1}]^T$ to the Cartesian coordinate $\mathbf{x}$. In (10), the variable r defines the radius (i.e., the distance to the origin) and the other $M-1$ variables $\{\theta_m; m = 1, 2, \ldots, M-1\}$ define the angle. Fig. 2(a) shows a 3-D example of the traditional spherical coordinate system.

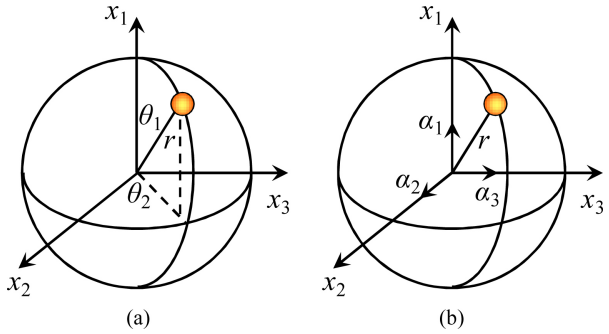


Fig. 2. 3-D example compares the difference between (a) traditional spherical coordinate system and (b) proposed spherical coordinate system.

In order to implement Gibbs sampling in an M-dimensional spherical coordinate system, we must find the "spherical" representation of the M-dimensional Normal PDF in (1). We need to transform the M-dimensional Normal PDF from the Cartesian coordinate system to the spherical coordinate system. While such a transformation can be easily found for low-dimensional cases (i.e., M is small), it becomes computationally expensive or even impossible as the dimensionality (i.e., M) increases [20].

Motivated by this observation, we define the spherical coordinate system in an alternative way so that the M-dimensional Normal PDF can be easily specified in our proposed spherical coordinate system. We write the Cartesian coordinate $\mathbf{x}$ as

$$x_1 = r \cdot \frac{\alpha_1}{\|\boldsymbol{\alpha}\|_2} \quad x_2 = r \cdot \frac{\alpha_2}{\|\boldsymbol{\alpha}\|_2} \quad \cdots \quad x_M = r \cdot \frac{\alpha_M}{\|\boldsymbol{\alpha}\|_2} \tag{11}$$

where $\boldsymbol{\alpha}$ equals $[\alpha_1\ \alpha_2 \ldots \alpha_M]^T$ and $||\cdot||_2$ denotes the L2-norm of a vector. In (11), the variable r defines the radius, which is similar to (10). The other M variables $\{\alpha_m; m = 1, 2, \ldots, M\}$ define the orientation of the vector $\mathbf{x}$. Fig. 2(b) shows a 3-D example of the proposed spherical coordinate system.

Comparing Fig. 2(a) and (b) reveals an important difference between the traditional spherical coordinate system and the proposed spherical coordinate system. While the traditional spherical coordinate system uses M variables (i.e., r and $\boldsymbol{\theta}$) to define a spatial location in the M-dimensional space, the proposed spherical coordinate system requires $M+1$ variables (i.e., r and $\boldsymbol{\alpha}$) to define the same spatial location. The $M+1$ variables used by our proposed spherical coordinate system are redundant. For a given Cartesian coordinate $\mathbf{x}$, it is not possible to uniquely determine the values of r and $\boldsymbol{\alpha}$. In other words, the mapping from $\mathbf{x}$ to r and $\boldsymbol{\alpha}$ is not unique. The variables r and $\boldsymbol{\alpha}$ do not form a basis of the M-dimensional space. Hence, the space spanned by r and $\boldsymbol{\alpha}$ is not formally an M-dimensional spherical coordinate system. However, by introducing the new variables r and $\boldsymbol{\alpha}$, we can easily find the joint PDF of r and $\boldsymbol{\alpha}$ to make the variables $\{x_m; m = 1, 2, \ldots, M\}$ in (11) mutually independent and standard Normal, thereby facilitating an easy implementation of the proposed Gibbs sampling method. In what follows, we will derive the joint probability distribution for r and $\boldsymbol{\alpha}$ in detail.

Remember that the variable r represents the radius, and hence it is equal to

$$r = \sqrt{x_1^2 + x_2^2 + \cdots + x_M^2}. \tag{12}$$

Since the random variables $\{x_m;\ m = 1, 2, \ldots, M\}$ are mutually independent and standard Normal as shown in (1), it is easy to verify that the random variable r follows the Chi distribution with M degrees of freedom [20]

$$f(r) = \frac{2 \cdot r^{M-1} \cdot \exp\left(-0.5 \cdot r^2\right)}{\sqrt{2^M} \cdot \Gamma(0.5 \cdot M)} \tag{13}$$

where $\Gamma(\cdot)$ denotes the Gamma function.

Next, we consider the vector $\boldsymbol{\alpha}$ that defines the orientation. By studying (11) we would have two important observations. First, the orientation of the vector $\mathbf{x}$ is uniquely determined by $\boldsymbol{\alpha}/||\boldsymbol{\alpha}||_2$. It is independent of the length of the vector $\boldsymbol{\alpha}$ (i.e., $||\boldsymbol{\alpha}||_2$). Second, since the random variables $\{x_m;\ m = 1, 2, \ldots, M\}$ in (11) are mutually independent and standard Normal, the orientation $\boldsymbol{\alpha}/||\boldsymbol{\alpha}||_2$ must be uniformly distributed. The unit length vector $\boldsymbol{\alpha}/||\boldsymbol{\alpha}||_2$ must take any possible orientation with equal probability. Based on these two observations, we define the following PDF for $\boldsymbol{\alpha}$:

$$f(\boldsymbol{\alpha}) = \prod_{m=1}^{M} \left[\frac{1}{\sqrt{2\pi}} \cdot \exp\left(-\frac{\alpha_m^2}{2}\right)\right]. \tag{14}$$

In other words, $\{\alpha_m; m = 1, 2, \ldots, M\}$ are mutually independent and standard Normal. It is well known that the multivariate Normal distribution in (14) is "spherically" symmetric [20]. Hence, all points on a sphere $||\boldsymbol{\alpha}||_2 = r$, where r is the radius of the sphere, have equal probability of occurring. Sampling $\boldsymbol{\alpha}$ based on (14) and normalizing $\boldsymbol{\alpha}$ by its L2-norm $||\boldsymbol{\alpha}||_2$ allow us to generate a uniformly distributed orientation in the M-dimensional space [17].

Finally, we combine $f(r)$ in (13) and $f(\boldsymbol{\alpha})$ in (14) together based on the fact that the radius and the orientation of the multivariate Normal distribution in (1) should be mutually independent. Hence, the joint PDF $f(r, \boldsymbol{\alpha})$ is equal to

$$f(r, \boldsymbol{\alpha}) = \frac{2 \cdot r^{M-1} \cdot \exp\left(-0.5 \cdot r^2\right)}{\sqrt{2^M} \cdot \Gamma(0.5 \cdot M)} \cdot \prod_{m=1}^{M} \left[\frac{1}{\sqrt{2\pi}} \cdot \exp\left(-\frac{\alpha_m^2}{2}\right)\right]. \tag{15}$$

Theorem 1 formally proves that given the joint PDF $f(r, \boldsymbol{\alpha})$ in (15), the random variables $\{x_m; m = 1, 2, \ldots, M\}$ defined in (11) are mutually independent and standard Normal. The detailed proof of Theorem 1 can be found in the Appendix.

Theorem 1: Given the random variables $\{x_m; m = 1, 2, \ldots, M\}$ defined in (11) where r and $\boldsymbol{\alpha}$ follow the joint PDF $f(r, \boldsymbol{\alpha})$ in (15), the joint PDF $f(\mathbf{x})$ is a multivariate Normal distribution represented by (1).

Given the aforementioned definition of r and $\boldsymbol{\alpha}$, importance sampling can be applied to estimate the failure probability in the proposed spherical coordinate system. Here, the estimated failure probability in (7) can be rewritten as follows:

$$\tilde{P}_f^{IS} = \frac{1}{N} \cdot \sum_{n=1}^{N} \frac{I\left[r^{(n)}, \boldsymbol{\alpha}^{(n)}\right] \cdot f\left[r^{(n)}, \boldsymbol{\alpha}^{(n)}\right]}{g\left[r^{(n)}, \boldsymbol{\alpha}^{(n)}\right]} \tag{16}$$

where $I(r, \boldsymbol{\alpha})$ and $g(r, \boldsymbol{\alpha})$ represent the indicator function and the distorted PDF in the spherical coordinate system, respectively, $[r^{(n)}\ \boldsymbol{\alpha}^{(n)}]^T$ stands for the nth sampling point drawn from $g(r, \boldsymbol{\alpha})$, and N denotes the total number of samples. Similar to (8), the optimal PDF $g(r, \boldsymbol{\alpha})$ leading to maximum estimation accuracy is as follows:

$$g^{OPT}(r, \boldsymbol{\alpha}) = \frac{I(r, \boldsymbol{\alpha}) \cdot f(r, \boldsymbol{\alpha})}{P_f}. \tag{17}$$

Given (16) and (17), Algorithm 2 can be formulated to perform Gibbs sampling in the spherical coordinate system. Similar to Algorithm 1, Algorithm 2 samples a 1-D conditional PDF $g^{OPT}(r \mid \boldsymbol{\alpha})$ or $g^{OPT}(\alpha_m \mid r, \boldsymbol{\alpha}_{\backslash m})$, where $\boldsymbol{\alpha}_{\backslash m}$ denotes the vector $\boldsymbol{\alpha}$ with α_m removed, at each iteration step.

To intuitively illustrate how the spherical coordinate implementation (i.e., Algorithm 2) works, we consider a simple 2-D example with two independent random variables x_1 and x_2 that follow the joint Normal distribution in (1). In this example, we simply assume the following failure region:

$$\Omega = \left\{(x_1, x_2) \quad \text{where} \quad x_1 \geq 0 \quad \text{and} \quad x_2 \geq 0\right\}. \tag{18}$$

When applying Algorithm 2 to generate Gibbs samples, we iteratively sample the random variable r defining the radius and the random variables α_1 and α_2 defining the orientation. Fig. 3 shows the scatter plots of x_1 and x_2 when sampling the conditional distribution $g^{OPT}(\alpha_1 \mid r, \alpha_2)$.

Studying Fig. 3 reveals two important observations. First, when the conditional distribution $g^{OPT}(\alpha_1 \mid r, \alpha_2)$ is sampled, the resulting random samples are distributed over a 2-D arc. Second, the length of the arc where the random samples are distributed depends on the value of α_2. In this example, the random samples can spread over a long arc if α_2 is small. This observation is consistent with the mapping defined in

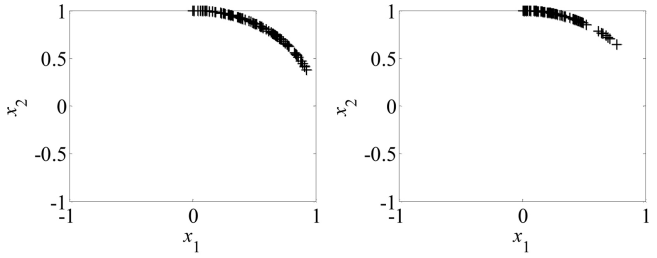


Fig. 3. Scatter plots with 100 random samples are shown for x_1 and x_2 when sampling the conditional distribution $g^{OPT}(\alpha_1 \mid r, \alpha_2)$ for the failure region defined in (18). (a) $r = 1$ and $\alpha_2 = 1$. (b) $r = 1$ and $\alpha_2 = 3$.

Algorithm 2 Gibbs sampling in spherical coordinate system

1. Start from an $(M + 1)$-dimensional PDF $g^{OPT}(r, \alpha_1, \alpha_2, \ldots, \alpha_M)$.
2. Select an initial starting point $[r^{(1)}\ \alpha_1^{(1)}\ \alpha_2^{(1)}\ \ldots\ \alpha_M^{(1)}]^T$.
3. For $t = 1, 2, \ldots$
4. Draw $r^{(t+1)}$ from the conditional PDF $g^{OPT}[r \mid \alpha_1^{(t)}, \ldots, \alpha_M^{(t)}]$ to create a new sampling point $[r^{(t+1)}\ \alpha_1^{(t)} \ldots\ \alpha_M^{(t)}]^T$.
5. For $m = 1, 2, \ldots, M$
6. Draw $\alpha_m^{(t+1)}$ from the conditional PDF $g^{OPT}[\alpha_m | r^{(t+1)}, \alpha_1^{(t+1)}, \ldots, \alpha_{m-1}^{(t+1)}, \alpha_{m+1}^{(t)}, \ldots, \alpha_M^{(t)}]$ to create a new sampling point $[r^{(t+1)}\ \alpha_1^{(t+1)}\ \ldots \alpha_m^{(t+1)}\ \alpha_{m+1}^{(t)}\ \ldots \alpha_M^{(t)}]^T$.
7. End For
8. End For

(11). Since the conditional PDF $g^{OPT}(\alpha_1 \mid r, \alpha_2)$ follows the same truncated Normal distribution in both cases where $\alpha_2 = 1$ and $\alpha_2 = 3$, respectively, the variation range of x_1 defined by (11) is large if the value of α_2 is small.

Unlike the Cartesian coordinate implementation where only one of the random variables (i.e., either x_1 or x_2) can vary at each iteration step, the proposed spherical coordinate implementation allows multiple random variables (i.e., both x_1 and x_2) to vary simultaneously over an arc, as shown in Fig. 3. Remember that x_1 and x_2 are independent and standard Normal, as defined in (1). It implies that the arc shown in Fig. 3 represents a contour line of the joint probability density function $f(x_1, x_2)$ on which the original random variables x_1 and x_2 are distributed with equal probability. In a general M-dimensional case, the proposed spherical coordinate implementation allows us to sample the random variables $\{x_m; m = 1, 2, \ldots, M\}$ over an M-dimensional contour line. Such a unique property facilitates the spherical coordinate implementation to offer superior accuracy over the Cartesian coordinate implementation in several cases, as will be demonstrated by the experimental results in Section V.

Finally, it is worth emphasizing that directly sampling the multidimensional joint PDF $g^{OPT}(\mathbf{x})$ or $g^{OPT}(r, \boldsymbol{\alpha})$ can be extremely difficult. By using Gibbs sampling, we only need to sample the 1-D conditional PDF $g^{OPT}(x_m \mid \mathbf{x}_{\backslash \mathbf{m}})$, $g^{OPT}(r \mid \boldsymbol{\alpha})$ or $g^{OPT}(\alpha_m \mid r, \boldsymbol{\alpha}_{\backslash m})$. As will be demonstrated in Section IV-A, such 1-D sampling can be efficiently implemented by using an inverse-transform method [22], even if the indicator function and, hence, the PDF $g^{OPT}(\mathbf{x})$ in (8) or $g^{OPT}(r, \boldsymbol{\alpha})$ in (17) are not explicitly known. In addition, a model-based optimization can be applied to determine a good starting point $[x_1^{(1)}\ x_2^{(1)}\ \ldots\ x_M^{(1)}]^T$ or $[r^{(1)}\ \alpha_1^{(1)}\ \alpha_2^{(1)}\ \ldots\ \alpha_M^{(1)}]^T$ so that the proposed Gibbs sampling algorithm converges quickly. All these implementation details will be discussed in Section IV.

IV. Implementation Details

The proposed Gibbs sampling method is efficiently implemented with several important techniques, including: 1) 1-D inverse-transform sampling; 2) initial starting point selection; and 3) two-stage Monte Carlo analysis. In what follows, we will discuss these implementation issues in detail.

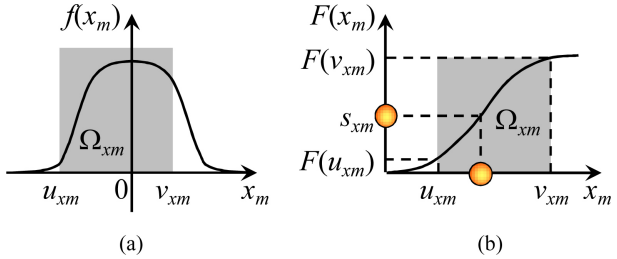
A. 1-D Inverse-Transform Sampling

During each iteration step of Gibbs sampling, one of the random variables is sampled from the 1-D conditional PDF $g^{OPT}(x_m|\mathbf{x}_{\backslash m})$, $g^{OPT}(r \mid \boldsymbol{\alpha})$, or $g^{OPT}(\alpha_m|r, \boldsymbol{\alpha}_{\backslash m})$. Such 1-D sampling can be efficiently performed by using an inverse-transform method [22]. To derive the 1-D sampling algorithm used in this paper, we first consider the scenario of sampling the random variable x_m from the conditional PDF $g^{OPT}(x_m|\mathbf{x}_{\backslash m})$. In this case, we write $g^{OPT}(x_m|\mathbf{x}_{\backslash m})$ as follows:

$$g^{OPT}\left(x_m|\mathbf{x}_{\backslash m}\right) = \frac{g^{OPT}\left(x_m, \mathbf{x}_{\backslash m}\right)}{g^{OPT}\left(\mathbf{x}_{\backslash m}\right)} = \frac{g^{OPT}\left(\mathbf{x}\right)}{g^{OPT}\left(\mathbf{x}_{\backslash m}\right)}. \tag{19}$$

Substituting (8) into (19) yields

$$g^{OPT}\left(x_m|\mathbf{x}_{\backslash m}\right) = \frac{I\left(\mathbf{x}\right) \cdot f\left(\mathbf{x}\right)}{P_f \cdot g^{OPT}\left(\mathbf{x}_{\backslash m}\right)}. \tag{20}$$



(a) (b)

Fig. 4. (a) Single continuous failure region Ω_{xm} is denoted by the gray area. (b) Random variable x_m is sampled by the inverse-transform method based on the CDF $F(x_m)$.

Since the indicator function $I(\mathbf{x})$ can be equivalently written as $I(x_m, \mathbf{x}_{\backslash m})$ and the joint PDF $f(\mathbf{x})$ equals the marginal PDF $f(\mathbf{x}_{\backslash m})$ multiplied by the conditional PDF $f(x_m \mid \mathbf{x}_{\backslash m})$, we have

$$\begin{aligned} g^{OPT}\left(x_m|\mathbf{x}_{\backslash m}\right) &= \frac{I\left(x_m, \mathbf{x}_{\backslash m}\right) \cdot f\left(\mathbf{x}_{\backslash m}\right) \cdot f\left(x_m|\mathbf{x}_{\backslash m}\right)}{P_f \cdot g^{OPT}\left(\mathbf{x}_{\backslash m}\right)} \\ &= \frac{f\left(\mathbf{x}_{\backslash m}\right)}{P_f \cdot g^{OPT}\left(\mathbf{x}_{\backslash m}\right)} \cdot I\left(x_m, \mathbf{x}_{\backslash m}\right) \cdot f\left(x_m|\mathbf{x}_{\backslash m}\right). \end{aligned} \tag{21}$$

The random variables $\{x_m;\ m = 1, 2, \ldots, M\}$ are mutually independent as shown in (1) and, hence, we have $f(x_m|\mathbf{x}_{\backslash m}) = f(x_m)$. Equation (21) can be rewritten as follows:

$$g^{OPT}\left(x_m|\mathbf{x}_{\backslash m}\right) = \frac{f\left(\mathbf{x}_{\backslash m}\right)}{P_f \cdot g^{OPT}\left(\mathbf{x}_{\backslash m}\right)} \cdot I\left(x_m, \mathbf{x}_{\backslash m}\right) \cdot f\left(x_m\right). \tag{22}$$

By studying (22) we would have three important observations. First, $g^{OPT}(x_m|\mathbf{x}_{\backslash m})$ is linearly proportional to the probability distribution $f(x_m)$. Second, $g^{OPT}(x_m|\mathbf{x}_{\backslash m})$ is nonzero if and only if the variable x_m, combined with $\mathbf{x}_{\backslash m}$, sits in the failure region. Third, the term $f(\mathbf{x}_{\backslash m})/P_f/g^{OPT}(\mathbf{x}_{\backslash m})$ in (22) is a constant, given any fixed value of $\mathbf{x}_{\backslash m}$. For these reasons, when sampling the 1-D conditional PDF $g^{OPT}(x_m|\mathbf{x}_{\backslash m})$, we should draw random samples from the failure region based on the PDF $f(x_m)$.

Toward this goal, we consider the 1-D failure region shown in Fig. 4(a) where the symbol Ω_{xm} represents the failure region over the variable x_m, while all other random variables are set to the given value $\mathbf{x}_{\backslash m}$. Note that the failure region Ω_{xm} is represented as a 1-D interval $[u_{xm}, v_{xm}]$. In other words, we assume that there is only one single continuous and bounded failure region where u_{xm} and v_{xm} are the left and right boundaries, respectively.

In practice, the failure region is likely to be continuous, if we consider only one failure mechanism at one time, similar to other previous works [8]–[10], [14]. For instance, taking read noise margin (RNM) as an example, read failure due to negative RNM can occur when reading either zero or one from the SRAM cell. If the topology of the SRAM cell is symmetric, there will be two discontinuous failure regions associated with reading zero and one, respectively [19]. However, if we consider the case of reading either zero or one at one time, there is only a single failure region. Once the failure rate associated with the single failure region is estimated, we can multiply its value by two to determine the total failure rate corresponding to both failure regions.

On the other hand, even if the actual failure region may not be bounded, we can bound the high-probability failure region

by constraining the random variable x_m within a given range. For instance, if the random variable x_m is standard Normal as shown in (1), it is possible to constrain x_m within the interval $[-\zeta, +\zeta]$ where ζ is a positive constant. As long as ζ is sufficiently large (e.g., ζ = 8–10), the probability for x_m to fall outside the interval $[-\zeta, +\zeta]$ is small and, hence, the approximation error is negligible.

Based on the assumption that Ω_{xm} is continuous and bounded, we first find the left boundary u_{xm} and the right boundary v_{xm} by performing binary search over the variable x_m with all other random variables set to the given value $\mathbf{x}_{\backslash m}$. Once u_{xm} and v_{xm} are known, we need to draw a random sample from the interval $[u_{xm}, v_{xm}]$. It is important to note that the PDF $g^{OPT}(x_m \,|\mathbf{x}_{\backslash m})$ in (22) represents a truncated Normal distribution defined by $f(x_m)$ over the interval $[u_{xm}, v_{xm}]$. It cannot be directly sampled by using a random number generator. The inverse-transform method samples $g^{OPT}(x_m \,|\mathbf{x}_{\backslash m})$ based on the cumulative distribution function (CDF) $F(x_m)$.

To apply the inverse-transform method, we first calculate the CDF $F(x_m)$ where x_m follows a standard Normal distribution. Next, we sample a new random variable s_{xm} that is uniformly distributed over the interval $[F(u_{xm}), F(v_{xm})]$, as shown in Fig. 4(b). Finally, we map the sampling point s_{xm} back to x_m based on the inverse CDF

$$x_m = F^{-1}(s_{xm}) \tag{23}$$

where $F^{-1}(s_{xm})$ is the inverse function of $F(x_m)$. It can be proven that the sampling point x_m generated by the inverse-transform method follows the statistical distribution $g^{OPT}(x_m \,|\mathbf{x}_{\backslash m})$ in (22) [22].

The aforementioned inverse-transform method can be similarly applied to sample the other two 1-D conditional PDFs $g^{OPT}(r|\boldsymbol{\alpha})$ and $g^{OPT}(\alpha_m|r, \boldsymbol{\alpha}_{\backslash m})$. Similar to (22), it is easy to verify the following two equations:

$$g^{OPT}(r|\boldsymbol{\alpha}) = \frac{f(\boldsymbol{\alpha})}{P_f \cdot g^{OPT}(\boldsymbol{\alpha})} \cdot I(r, \boldsymbol{\alpha}) \cdot f(r) \tag{24}$$

$$g^{OPT}\left(\alpha_m|r, \boldsymbol{\alpha}_{\backslash m}\right) = \frac{f\left(r, \boldsymbol{\alpha}_{\backslash m}\right)}{P_f \cdot g^{OPT}\left(r, \boldsymbol{\alpha}_{\backslash m}\right)} \cdot I(r, \boldsymbol{\alpha}) \cdot f(\alpha_m). \tag{25}$$

In (24) and (25), $f(r)$ is a Chi distribution with M degrees of freedom and $f(\alpha_m)$ is a standard Normal distribution. To sample $g^{OPT}(r \mid \boldsymbol{\alpha})$ and $g^{OPT}(\alpha_m \mid r, \boldsymbol{\alpha}_{\backslash m})$, we first apply binary search to find the 1-D failure regions $[u_r, v_r]$ for the variable r and $[u_{\alpha m}, v_{\alpha m}]$ for the variable α_m. Next, we sample the new random variables s_r and $s_{\alpha m}$ from the uniform distributions over $[F(u_r), F(v_r)]$ and $[F(u_{\alpha m}), F(v_{\alpha m})]$, respectively. Finally, we map s_r and $s_{\alpha m}$ back to r and α_m based on the inverse CDFs

$$r = F^{-1}(s_r) \tag{26}$$

$$\alpha_m = F^{-1}(s_{\alpha m}) \tag{27}$$

where $F^{-1}(s_r)$ and $F^{-1}(s_{\alpha m})$ are the inverse functions of the CDFs $F(r)$ and $F(\alpha_m)$, respectively.

Algorithm 3 summarizes the major steps of the inverse-transform method. It is important to emphasize that the computational cost of Algorithm 3 is dominated by Step 2, where multiple transistor-level simulations are required to find the left and right boundaries of the 1-D failure region. All other steps do not involve transistor-level simulations and, hence, can be computed with a low computational cost.

Algorithm 3 1-D inverse-transform sampling

1. Start from the joint PDF $f(\mathbf{x})$ in (1) or $f(r, \boldsymbol{\alpha})$ in (15), a given random variable (i.e., x_m, r or α_m) for sampling, and the sampled values of all other random variables that should be fixed during the current iteration step of Gibbs sampling.
2. Apply binary search to find the left boundary (i.e., u_{xm}, u_r or $u_{\alpha m}$) and the right boundary (i.e., v_{xm}, v_r or $v_{\alpha m}$) for the 1-D failure region of the random variable that should be sampled (i.e., x_m, r or α_m).
3. Draw one random sample (i.e., s_{xm}, s_r or $s_{\alpha m}$) from the uniform distribution over the 1-D failure region (i.e., $[F(u_{xm}), F(v_{xm})]$, $[F(u_r), F(v_r)]$ or $[F(u_{\alpha m}), F(v_{\alpha m})]$).
4. Map the random sample s_{xm}, s_r or $s_{\alpha m}$ to x_m, r or α_m based on (23), (26) or (27) where $F^{-1}(s_{xm})$ and $F^{-1}(s_{\alpha m})$ are the inverse CDFs of a standard Normal distribution and $F^{-1}(s_r)$ is the inverse CDF of a Chi distribution with M degrees of freedom.

B. Initial Starting Point Selection

To efficiently implement the proposed Gibbs sampling method, a good initial starting point should be appropriately selected to speed up the convergence of Algorithm 1 and/or Algorithm 2. During Gibbs sampling, as each sample is drawn from a 1-D conditional PDF, the current sampling point is correlated to the previous sampling point. In other words, the random samples generated by Gibbs sampling depend on the initial starting point. It has been proven by the statistics community that starting from an initial point, a number of random samples generated at the beginning of the Gibbs sampling process may not follow the given probability distribution (i.e., $g^{OPT}(\mathbf{x})$ for Algorithm 1 or $g^{OPT}(r, \boldsymbol{\alpha})$ for Algorithm 2) [22]. These samples form the warm-up interval of Gibbs sampling.

The sampling points inside the warm-up interval should be abandoned, since they do not represent the actual probability distribution that we want to sample. In practice, the length of the warm-up interval should be minimized; otherwise, generating a large number of random samples inside the warm-up interval can waste a large amount of computational time. In this paper, we propose to reduce (or even almost eliminate) the warm-up interval by appropriately selecting a good initial starting point. In what follows, we first discuss the proposed initial point selection scheme for Cartesian coordinate systems.

Remember that our objective is to draw random samples from the failure region that is most likely to occur, as shown in (8). Hence, the initial starting point $[x_1^{(1)}\ x_2^{(1)} \ldots x_M^{(1)}]^T$ should meet the following two criteria: 1) $[x_1^{(1)}\ x_2^{(1)} \ldots x_M^{(1)}]^T$ is inside the failure region; and 2) the PDF of process variations, i.e., $f(\mathbf{x})$, takes a large value at $[x_1^{(1)}\ x_2^{(1)} \ldots x_M^{(1)}]^T$. These two requirements can be mathematically translated to the following

optimization problem:

$$\begin{aligned} \underset{x_1^{(1)},\cdots,x_M^{(1)}}{\text{maximize}} \quad & f\left[x_1^{(1)}, x_2^{(1)}, \cdots, x_M^{(1)}\right] \\ \text{s.t.} \quad & \left[\begin{array}{cccc} x_1^{(1)} & x_2^{(1)} & \cdots & x_M^{(1)} \end{array}\right]^T \in \Omega. \end{aligned} \tag{28}$$

In (28), we want to find the failure point that is most likely to occur. Since the M-dimensional random variable $\mathbf{x}$ is modeled as a multivariate Normal distribution in (1), the optimization in (28) is equivalent to

$$\begin{aligned} \underset{x_1^{(1)},\cdots,x_M^{(1)}}{\text{minimize}} \quad & \left[x_1^{(1)}\right]^2 + \left[x_2^{(1)}\right]^2 + \cdots + \left[x_M^{(1)}\right]^2 \\ \text{s.t.} \quad & \left[\begin{array}{cccc} x_1^{(1)} & x_2^{(1)} & \cdots & x_M^{(1)} \end{array}\right]^T \in \Omega. \end{aligned} \tag{29}$$

Equation (29) aims to find the failure point that is closest to the origin $\mathbf{x} = \mathbf{0}$. It is similar to the norm minimization approach proposed in [10].

Note that solving the optimization problem in (29) is not trivial, since the failure region Ω is not explicitly known. To address this issue, we propose to approximate the performance of interest (e.g., read margin, write margin) as a linear or quadratic model of the M-dimensional random variable $\mathbf{x}$. Once the model is available, the optimization in (29) can be solved by either quadratic programming (for linear performance model) or semidefinite programming (for quadratic performance model) [18]. The detailed algorithm for performance modeling and optimization can be found in [18]. It is important to mention that even though the linear or quadratic performance models may not be highly accurate to cover a large variation space, we can still obtain a good starting point required by the proposed Gibbs sampling algorithm. Note that our goal is not to exactly solve (29). Instead, we only want to find an approximate solution that can be used by Gibbs sampling to further explore the variation space with high failure probability.

Next, we will further extend the aforementioned initial point selection method to spherical coordinate systems. In this case, once the Cartesian coordinate $[x_1^{(1)}\ x_2^{(1)} \ldots x_M^{(1)}]^T$ is determined for the initial starting point, we need to map it to the spherical coordinate $[r^{(1)}\ \alpha_1^{(1)}\ \alpha_2^{(1)} \ldots \alpha_M^{(1)}]^T$. As shown in (12), the value of $r^{(1)}$ can be easily calculated as follows:

$$r^{(1)} = \sqrt{\left[x_1^{(1)}\right]^2 + \left[x_2^{(1)}\right]^2 + \cdots + \left[x_M^{(1)}\right]^2}. \tag{30}$$

To calculate $\{\alpha_m^{(1)}; m = 1, 2, \ldots, M\}$, (11) can be rewritten as follows:

$$\begin{aligned} \alpha_1^{(1)} = \left\|\boldsymbol{\alpha}^{(1)}\right\|_2 \cdot \frac{x_1^{(1)}}{r^{(1)}} \quad \alpha_2^{(1)} = \left\|\boldsymbol{\alpha}^{(1)}\right\|_2 \cdot \frac{x_2^{(1)}}{r^{(1)}} \quad \cdots \\ \alpha_M^{(1)} = \left\|\boldsymbol{\alpha}^{(1)}\right\|_2 \cdot \frac{x_M^{(1)}}{r^{(1)}}. \end{aligned} \tag{31}$$

As discussed in Section III-B, the vector $\boldsymbol{\alpha}^{(1)}$ only determines the orientation of $\mathbf{x}^{(1)}$ and the length of $\boldsymbol{\alpha}^{(1)}$ can be arbitrary. In other words, the values of $\{\alpha_m^{(1)}; m = 1, 2, \ldots, M\}$ cannot be uniquely determined based on the Cartesian coordinate $\mathbf{x}^{(1)}$. For our Gibbs sampling application, since we aim to locate a failure point $[r^{(1)}\ \alpha_1^{(1)}\ \alpha_2^{(1)} \ldots \alpha_M^{(1)}]^T$ that is most likely to occur, we should determine $\{\alpha_m^{(1)}; m = 1, 2, \ldots, M\}$ by maximizing the PDF $f(\boldsymbol{\alpha})$ in (14). Even though there are multiple possible solutions of $\boldsymbol{\alpha}^{(1)}$ that are associated with the same Cartesian coordinate $\mathbf{x}^{(1)}$, we are interested in the solution that is most likely to occur (i.e., the maximum-likelihood solution). Based on this maximum-likelihood criterion and the fact that $\boldsymbol{\alpha}^{(1)}$ follows the multivariate Normal distribution defined in (14), the length of $\boldsymbol{\alpha}^{(1)}$ should be sufficiently small so that the PDF value $f[\boldsymbol{\alpha}^{(1)}]$ is large. Let $||\boldsymbol{\alpha}^{(1)}||_2 = \varepsilon$, where $\varepsilon \to 0$, and (31) becomes

$$\alpha_1^{(1)} = \varepsilon \cdot \frac{x_1^{(1)}}{r^{(1)}} \quad \alpha_2^{(1)} = \varepsilon \cdot \frac{x_2^{(1)}}{r^{(1)}} \quad \cdots \quad \alpha_M^{(1)} = \varepsilon \cdot \frac{x_M^{(1)}}{r^{(1)}}. \tag{32}$$

In practice, we find that $\varepsilon = 10^{-3}$–10^{-2} is a good choice based on our numerical experiments. Once ε is set and $\{x_m^{(1)}; m = 1, 2, \ldots, M\}$ and $r^{(1)}$ are found by (29) and (30), $\{\alpha_m^{(1)}; m = 1, 2, \ldots, M\}$ are uniquely determined by (32). Algorithm 4 summarizes the initial starting point selection scheme for both Cartesian and spherical coordinate systems.

Algorithm 4 Initial starting point selection

1. Start from the joint PDF $f(\mathbf{x})$ in (1).
2. Solve the optimization problem in (29) to find the Cartesian coordinate $\mathbf{x}^{(1)}$ by using the performance modeling technique described in [18].
3. Map the Cartesian coordinate $\mathbf{x}^{(1)}$ to the spherical coordinate $r^{(1)}$ and $\boldsymbol{\alpha}^{(1)}$ based on (30) and (32).

C. Two-Stage Monte Carlo Flow

To make the proposed Gibbs sampling algorithm of practical utility, there is one additional implementation issue that should be further addressed. Unlike the traditional Gibbs sampling algorithm used by the statistics community [16], [22] where the sampling PDF is given, we do not explicitly know the multidimensional joint PDF $g^{OPT}(\mathbf{x})$ in (8) or $g^{OPT}(r, \boldsymbol{\alpha})$ in (17) as the indicator function $I(\mathbf{x})$ or $I(r, \boldsymbol{\alpha})$ is unknown. In this case, even if we can generate the Gibbs samples by applying Algorithm 3, we cannot simply use the importance sampling formula in (7) or (16) to calculate the failure probability. Furthermore, as shown by both Algorithm 1 and Algorithm 2, multiple (typically 5–10) transistor-level simulations are required to perform binary search and generate a single Gibbs sample. It implies that applying Gibbs sampling to create many random samples can be expensive, since it requires repeatedly running a large number of transistor-level simulations.

For these reasons, once a set of (say, K) Gibbs samples are created, it is desired to "learn" the joint PDF (e.g., $g^{OPT}(\mathbf{x})$ in (8) for importance sampling in Cartesian coordinate systems) from these samples. As such, additional random sampling points can be directly drawn from the PDF that is learned without running binary search. Such a strategy would help to make the proposed Gibbs sampling technique applicable to our application of SRAM failure rate prediction. It also further reduces the computational cost and/or improves the prediction accuracy.

Toward this goal, we propose to adopt a two-stage Monte Carlo flow consisting of two sequential steps. First, Gibbs sampling is applied to generate K random samples inside the failure region. Next, at the second stage, we approximate the

Algorithm 5 Two-stage Monte Carlo flow

1. Start from a joint PDF $f(\mathbf{x})$, a given value of K (i.e., the number of Gibbs samples for the first stage), and a given value of N (i.e., the number of random samples for the second stage).
2. Apply Algorithm 1 (for Cartesian coordinate systems only), Algorithm 2 (for spherical coordinate systems only), Algorithm 3 (for both Cartesian and spherical coordinate systems), and Algorithm 4 (for both Cartesian and spherical coordinate systems) to generate K Gibbs sampling points.
3. If the Gibbs samples are generated in a spherical coordinate system, map them to the Cartesian coordinates by using (11).
4. Calculate the mean value and the covariance matrix of these K Gibbs samples in the Cartesian coordinate system. Determine a multivariate Normal distribution $g^{NOR}(\mathbf{x})$ to approximate the joint PDF for importance sampling.
5. Generate N random samples from the multivariate Normal distribution $g^{NOR}(\mathbf{x})$.
6. Calculate the failure rate from these N samples by using (33).

optimal PDF $g^{OPT}(\mathbf{x})$ as a multivariate Normal distribution $g^{NOR}(\mathbf{x})$ where the mean value and the covariance matrix of $g^{NOR}(\mathbf{x})$ are calculated from the K Gibbs samples [23]. In our implementation, since the initial starting point is appropriately selected by Algorithm 4, we assume no warm-up interval and all Gibbs samples from the first stage are used to estimate the Normal distribution $g^{NOR}(\mathbf{x})$. The second stage is always implemented in Cartesian coordinate systems. If Algorithm 2 is used to generate Gibbs samples in a spherical coordinate system, these samples can be easily mapped to their Cartesian coordinates based on (11) before fitting the Normal distribution $g^{NOR}(\mathbf{x})$. Once $g^{NOR}(\mathbf{x})$ is known, we directly sample it to generate N random samples and estimate the failure rate from these N samples

$$\tilde{P}_f^{IS} = \frac{1}{N} \cdot \sum_{n=1}^{N} \frac{I\left[\mathbf{x}^{(n)}\right] \cdot f\left[\mathbf{x}^{(n)}\right]}{g^{NOR}\left[\mathbf{x}^{(n)}\right]}. \tag{33}$$

Algorithm 5 summarizes the major steps of the proposed two-stage Monte Carlo flow.

Algorithm 5 requires a user to specify the appropriate values of K (i.e., the number of Gibbs samples for the first stage) and N (i.e., the number of random samples for the second stage). These two parameters are often empirically chosen. Based on our experience, we set K and N in the order of 10^2–10^3 and 10^3–10^4, respectively, for our experimental examples in Section V.

While Algorithm 5 approximates the optimal PDF $g^{OPT}(\mathbf{x})$ as a multivariate Normal distribution, $g^{OPT}(\mathbf{x})$ can also be approximated as other non-Normal distributions such as Gaussian mixture distribution [23]. However, these non-Normal distributions often require more Gibbs samples to fit than a Normal distribution. In other words, we will have to increase the number of Gibbs samples in the first stage to accurately fit a non-Normal distribution. In this paper, we focus on the simple Normal approximation, as shown by Algorithm 5. The possible extension to other non-Normal distributions will be considered in our future research.

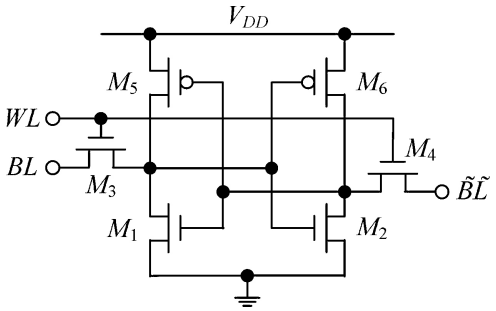


Fig. 5. 6-T SRAM cell is used as the test case to demonstrate the efficacy of the proposed Gibbs sampling method.

Finally, it should be noted that several traditional importance sampling techniques also draw random samples from a multivariate Normal distribution [8], [14]. However, unlike the traditional techniques that only estimate the mean value of the multivariate Normal distribution for importance sampling, we apply Gibbs sampling to optimally determine both the mean value and the covariance matrix for $g^{NOR}(\mathbf{x})$. Hence, the second-stage random sampling can converge quickly.

On the other hand, Algorithm 5 often requires more first-stage simulation runs than other traditional techniques [8], [14] in order to accurately estimate the covariance matrix for second-stage sampling. For this reason, the proposed method is expected to be more efficient than the traditional approaches, when the error of failure rate prediction must be sufficiently small (e.g., the relative error defined by the 99% confidence interval reaches 5%). In these cases, most traditional techniques require a large number of second-stage samples to achieve such a high prediction accuracy and, hence, the proposed two-stage Monte Carlo flow is preferred due to its fast convergence at the second stage, as will be demonstrated by our experimental results in Section V.

V. Numerical Examples

Fig. 5 shows the circuit schematic of a 6-T SRAM cell designed in a 90 nm CMOS process. In this section, the SRAM cell is used to demonstrate the efficacy of the proposed Gibbs sampling method. For testing and comparison purposes, four different importance sampling methods are implemented: 1) mixture importance sampling (MIS) [8]; 2) minimum-norm importance sampling (MNIS) [14]; 3) the proposed Gibbs sampling implemented for Cartesian coordinate systems (G-C); and 4) the proposed Gibbs sampling implemented for spherical coordinate systems (G-S) where we set $\varepsilon = 10^{-2}$ in (32) to generate the initial starting point by Algorithm 4. All these four methods employ a two-stage analysis flow. A multivariate Normal distribution $g^{NOR}(\mathbf{x})$ is first constructed from K random samples during the first stage and then N additional random samples are drawn from $g^{NOR}(\mathbf{x})$ to calcu-

A. Noise Margin

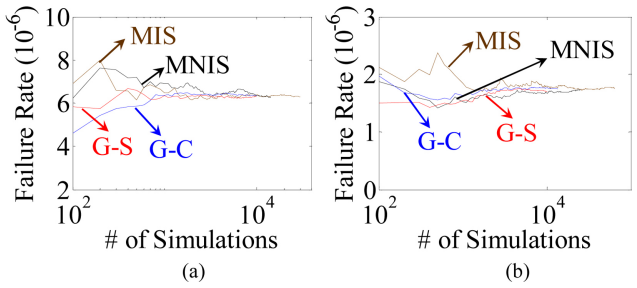


Fig. 6. Estimated failure probability is plotted as a function of the number of transistor-level simulations at the second stage. (a) RNM. (b) WNM.

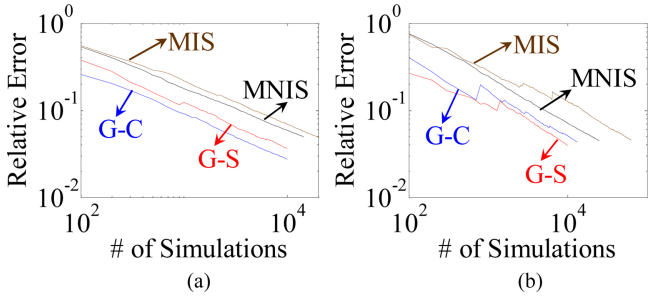


Fig. 7. Relative error of failure rate prediction (defined by 99% confidence interval) is plotted as a function of the number of transistor-level simulations at the second stage. (a) RNM. (b) WNM.

TABLE I
NUMBER OF REQUIRED SIMULATIONS FOR BOTH FIRST AND SECOND STAGES TO ACHIEVE 5% ERROR DEFINED BY 99% CONFIDENCE INTERVAL

	First Stage	Second Stage		Total	
		RNM	WNM	RNM	WNM
MIS [8]	5000	19 100	53 200	24 100	58 200
MNIS [14]	1000	14 200	20 100	15 200	21 100
G-C (proposed)	5000	3100	10 400	8100	15 400
G-S (proposed)	5000	5100	6800	10 100	11 800

late the failure probability at the second stage. All numerical experiments are run on a 2.53 GHz server with 16 GB memory.

A. Noise Margin

In this subsection, two performance metrics, RNM and write noise margin (WNM), are used to assess the stability of the SRAM cell. Both RNM and WNM must be greater than zero to ensure a stable SRAM cell. The importance sampling methods are applied to estimate the failure rate associated with RNM and WNM. The local V_{TH} mismatches of all transistors (i.e., six random variables $\{\Delta V_{TH1}, \Delta V_{TH2}, \ldots, \Delta V_{TH6}\}$) are considered for Monte Carlo analysis. These random variables are modeled as a joint Normal distribution.

To compare the difference between the four importance sampling algorithms, Fig. 6 shows the estimated failure probability as a function of the number of transistor-level simulations at the second stage. All four importance sampling methods yield the same failure probability, if the number of random samples is sufficiently large. In this example, the proposed Gibbs sampling methods (i.e., G-C and G-S) are more accurate than the other two traditional methods (i.e., MIS and MNIS) given the same number of random samples.

To quantitatively assess the accuracy of different importance sampling methods, Fig. 7 plots the relative prediction error as a function of the number of second-stage simulations. Here, the relative error is defined as the ratio of the 99% confidence interval over the estimated failure probability. Table I further compares the number of required simulations of both first and second stages to achieve 5% error. For both G-C and G-S, the first-stage simulations include two portions: 1) the simulations required for initial starting point selection; and 2) the simulations required for generating Gibbs samples. By studying Table I, we note that the proposed Gibbs sampling methods only require 8100–15 400 simulations in total, while MIS and MNIS require 15 200–58 200 simulations to achieve the same accuracy. In other words, G-C and G-S achieve 1.4–4.9× runtime speedup over MIS and MNIS in this example.

To further understand the advantages and limitations of the aforementioned importance sampling methods, Figs. 8–11 plot the random samples that are generated at the second stage. For illustration purposes, these figures only show the V_{TH} mismatches of two transistors that are critical to the performance of interest (i.e., ΔV_{TH1} and ΔV_{TH3} for RNM and ΔV_{TH3} and ΔV_{TH5} for WNM) for the SRAM cell in Fig. 5. Each random sample is labeled as "Pass" or "Fail." It indicates whether the performance of interest (i.e., RNM or WNM) associated with the random sample passes or fails the specification.

Studying Figs. 8–11 reveals three important observations. First, the failure points sampled by four different importance sampling methods fall into the same failure region. It, in turn, implies that all four importance sampling methods attempt to draw random samples from the same failure region in this example. Second, both MIS and MNIS do not accurately approximate the optimal PDF $g^{OPT}(\mathbf{x})$ for importance sampling. These two traditional methods only identify the mean value of $g^{OPT}(\mathbf{x})$, while the covariance matrix is completely ignored. Hence, a large number of random samples generated at the second stage do not fall into the failure region. The performance values associated with these random samples are labeled as "Pass" in Figs. 8 and 9. Third, the proposed Gibbs sampling algorithms (i.e., G-C and G-S) are able to accurately capture both the mean value and the covariance matrix of $g^{OPT}(\mathbf{x})$. For this reason, most sampling points generated by G-C and G-S appropriately cover the failure region of interest, as shown in Figs. 10 and 11.

The aforementioned three observations demonstrate that while all four importance sampling methods sample the same failure region in this example, G-C and G-S are able to generate failure points more efficiently than MIS and MNIS. This is the fundamental reason why G-C and G-S achieve superior accuracy over MIS and MNIS for failure rate prediction with the same number of simulations, as demonstrated by the results shown in Figs. 6, 7, and Table I.

Finally, it is worth mentioning that even though G-C and G-S yield similar accuracy in this example, they can lead to different results in a number of other cases. One of

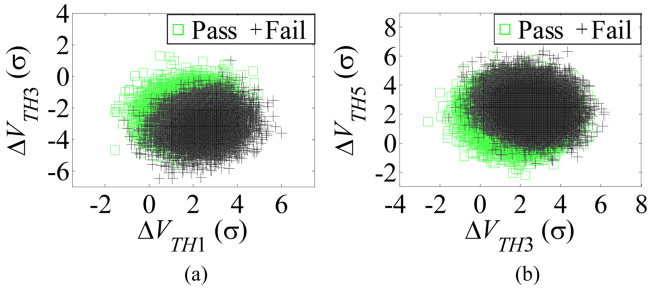


Fig. 8. Random samples generated by MIS (i.e., mixture importance sampling [8]) at the second stage are plotted for (a) RNM and (b) WNM.

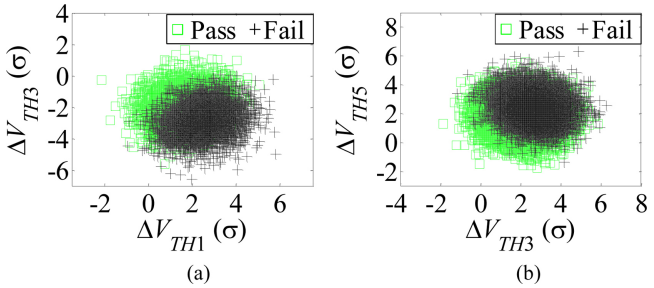


Fig. 9. Random samples generated by MNIS (i.e., minimum-norm importance sampling [14]) at the second stage are plotted for (a) RNM and (b) WNM.

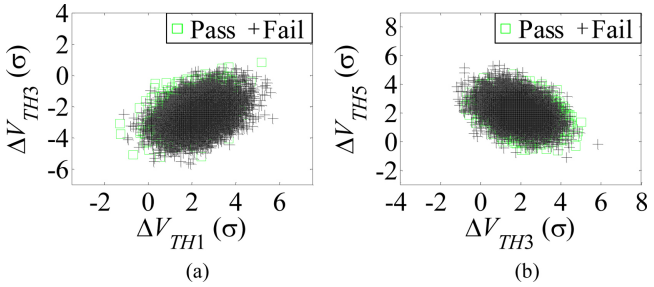


Fig. 10. Random samples generated by G-C (i.e., the proposed Gibbs sampling implemented for Cartesian coordinate systems) at the second stage are plotted for (a) RNM and (b) WNM.

these examples will be discussed in detail in the next subsection.

B. Read Current

In this subsection, we consider the read current of the SRAM cell as the performance of interest. Given the circuit schematic shown in Fig. 5, the read current is measured as the drain current of the transistor M_3, when the word line (i.e., *WL*) and both bit lines (i.e., *BL* and $\tilde{B}\tilde{L}$) are connected to the supply voltage V_{DD}. The read current directly impacts the discharge speed of bit lines during a read operation. Hence, it is an important performance metric related to access time failure. If the read current is greater than or equal to a predefined threshold, the SRAM cell is considered as "Pass."

We apply four different importance sampling methods to estimate the failure rate associated with the performance metric of read current. In this example, the local V_{TH} mismatches of the transistors M_1 and M_3 (i.e., two random variables $\{\Delta V_{TH1}, \Delta V_{TH3}\}$) are considered only due to the following reasons. First, the read current variation is dominated by the local

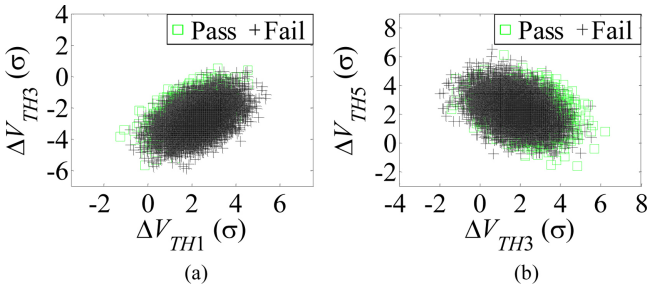


Fig. 11. Random samples generated by G-S (i.e., the proposed Gibbs sampling implemented for spherical coordinate systems) at the second stage are plotted for (a) RNM and (b) WNM.

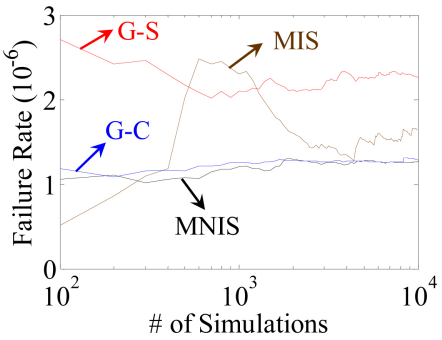


Fig. 12. Estimated failure probability of read current is plotted as a function of the number of transistor-level simulations at the second stage.

TABLE II
FAILURE PROBABILITY OF READ CURRENT ESTIMATED BY DIFFERENT IMPORTANCE SAMPLING METHODS

	Number of Simulations		Failure Rate	Relative Error
	First Stage	Second Stage		
MIS [8]	5×10^3	1×10^4	1.64×10^{-6}	28.1%
MNIS [14]	1×10^3	1×10^4	1.26×10^{-6}	44.7%
G-C (proposed)	5×10^3	1×10^4	1.29×10^{-6}	43.4%
G-S (proposed)	5×10^3	1×10^4	2.25×10^{-6}	1.32%
Brute-force MC	8.7×10^6		2.28×10^{-6}	–

V_{TH} mismatches of these two transistors. Second, considering ΔV_{TH1} and ΔV_{TH3} only (i.e., a 2-D variation space) facilitates us to make a comprehensive comparison of different importance sampling algorithms and, hence, fully understand their advantages and limitations based on the numerical results presented in this subsection.

Fig. 12 shows the estimated failure probability as a function of the number of transistor-level simulations at the second stage. Unlike the RNM and WNM examples in the previous sub-section where all four importance sampling methods eventually converge to the same failure probability, the proposed Gibbs sampling for spherical coordinate systems (i.e., G-S) results in a failure rate that is different from the other three methods (i.e., MIS, MNIS, and G-C). In order to verify the accuracy of these four importance sampling methods, we further implement a brute-force Monte Carlo analysis engine where 8.7 million random samples are directly drawn from the PDF of process variations, i.e., $f(\mathbf{x})$ in (1). The failure probability calculated by the brute-force Monte Carlo analysis

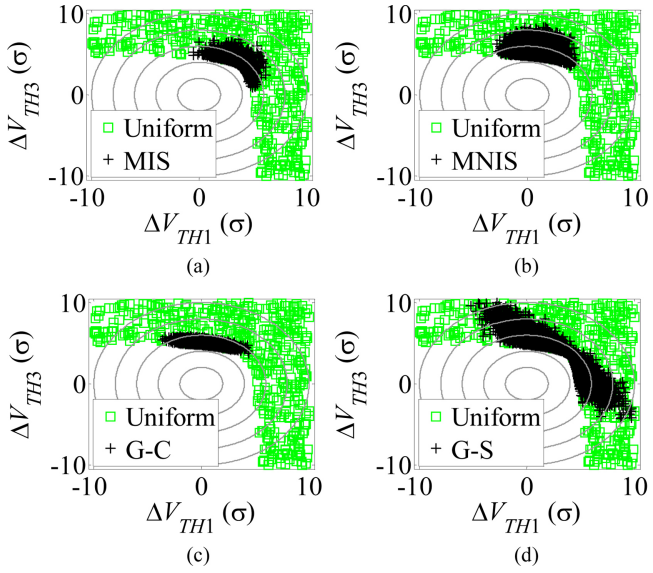


Fig. 13. 2-D failure region is plotted for the performance metric of read current. The gray circles denote the contour lines of the joint probability density function $f(\Delta V_{TH1}, \Delta V_{TH3})$ for the random variables ΔV_{TH1} and ΔV_{TH3}. Each green square represents a failure point that is randomly sampled from a 2-D uniform distribution. Each black cross denotes a failure point that is generated by the importance sampling methods at the second stage. (a) MIS [8]. (b) MNIS [14]. (c) G-C. (d) G-S.

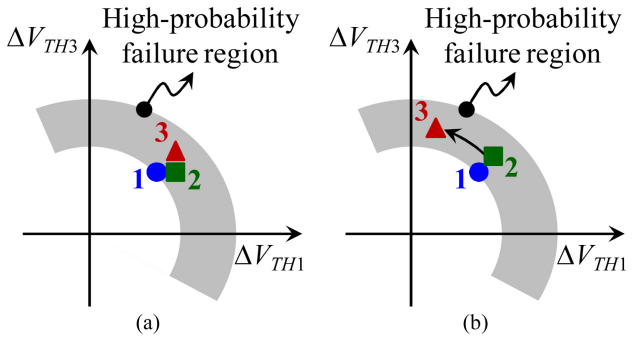


Fig. 14. First three Gibbs samples are conceptually illustrated to compare the difference between (a) G-C and (b) G-S. The gray area stands for the high-probability failure region. The blue circle (labeled as "1") indicates the initial starting point, the green square (labeled as "2") represents the second Gibbs sample, and the red triangle (labeled as "3") denotes the third Gibbs sample.

engine is used as the "golden" result to assess the accuracy of all importance sampling methods in this example.

Table II summarizes the failure probability estimated by the four importance sampling methods and the brute-force Monte Carlo analysis engine. On the basis of Table II, we notice that G-S is more accurate than the other three importance sampling methods (i.e., MIS, MNIS, and G-C), since the failure rate estimated by G-S is almost identical to that estimated by the brute-force Monte Carlo analysis engine with 8.7 million samples. It is important to note that MIS, MNIS, and G-C do not result in the correct failure rate, even though the number of random samples reaches 10 000 at the second stage, as shown in Fig. 12. In addition, once the number of second-stage simulations reaches 6000, collecting extra random samples for MIS, MNIS, and G-C does not help to quickly reduce the estimation error. In other words, the error associated with MIS, MNIS, and G-C cannot be efficiently reduced by increasing the number of samples at the second stage. While G-S accurately estimates the correct failure probability, MIS, MNIS, and G-C all fail to work in this example.

To fully understand the limitations of MIS, MNIS, and G-C, we uniformly sample the variation space to identify the failure region for the performance metric of read current. Since only the local V_{TH} mismatches of two transistors are considered in this example, the 2-D failure region can be easily found by the aforementioned uniform sampling, as shown in Fig. 13. Fig. 13 further plots the failure points generated by the four importance sampling methods at the second stage.

Several important observations can be obtained by studying the data in Fig. 13. First, the proposed G-S method is able to generate random samples to fully cover the high-probability failure region (i.e., the failure region that is close to the origin). Second, the other three importance sampling methods (i.e., MNS, MNIS, and G-C) cannot appropriately sample the high-probability failure region in this example. Instead, they only draw random samples from a small portion of the high-probability failure region, thereby underestimating the failure probability. If we further increase the number of random samples at the second stage, MNS, MNIS, and G-C can eventually converge to the correct failure rate. However, we expect that an extremely large number of random samples must be collected before these three methods can reach convergence.

In this example, MNS, MIS, and G-C all fail to work, since the high-probability failure region corresponds to an irregular, nonconvex shape, as shown in Fig. 13. When MNS and MNIS are applied, they only shift the mean value of the multivariate Normal distribution $f(\mathbf{x})$ in (1) to construct a new Normal distribution $g^{NOR}(\mathbf{x})$ for importance sampling. Since the covariance matrix is not appropriately estimated based on the high-probability failure region, both MNS and MNIS cannot correctly capture the failure region of interest.

To further study the difference between G-C and G-S, Fig. 14 conceptually illustrates the locations of the first three Gibbs samples generated by both methods. In this example, both G-C and G-S initially start from the same blue circle that is determined by Algorithm 4. The initial starting point sits in the high-probability failure region and it is close to the boundary of the failure region.

When G-C is applied to generate Gibbs samples, it first samples the random variable ΔV_{TH1}, while the other random variable ΔV_{TH3} is fixed. Since ΔV_{TH1} follows a Normal distribution, its probability density function $f(\Delta V_{TH1})$ exponentially decays, as ΔV_{TH1} moves away from zero. As a result, G-C yields a sample that is close to the boundary of the failure region (namely, close to the origin of the Cartesian coordinate system), as shown by the green square in Fig. 14(a). Note that the green square is extremely close to the blue circle. Next, G-C samples ΔV_{TH3}, while ΔV_{TH1} is fixed. For the same reason, the new sample is close to the boundary of the failure region, as shown by the red triangular in Fig. 14(a). The new Gibbs sample cannot move far away from the previous sample. Hence, the Gibbs samples iteratively generated by G-C are locally distributed in a small region. This conclusion can be fur-

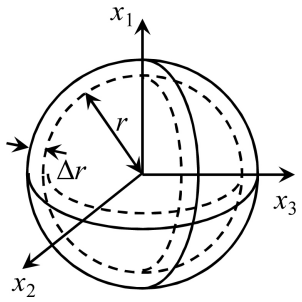


Fig. 15. Perturbation Δr, where $\Delta r \to 0$, is applied to the radius of the 3-D sphere $||\mathbf{x}||_2 = r$.

ther supported by Fig. 13(c), where the failure points generated at the second stage are distributed over a small local region.

On the other hand, G-S is able to successfully generate Gibbs samples to fully cover the high-probability failure region. To understand the reason, we need to take a close look at the sampling process associated with G-S. As shown in Fig. 14(b), G-S starts from the blue circle and it first samples over the radius to reach the green square. Next, G-S samples over the contour line of the joint probability density function $f(\Delta V_{TH1}, \Delta V_{TH3})$, similar to the 2-D example shown in Fig. 3. It results in a new sample that is far away from the initial starting point, as shown by the red triangular in Fig. 14(b). This is the fundamental reason why G-S does not get stuck within a small local region in this example. The aforementioned discussions demonstrate an important fact that even though both G-C and G-S rely on Gibbs sampling, they can lead to substantially different results due to their different implementations.

Finally, it is worth mentioning that the high-probability failure region is not continuous along the contour line of the joint probability density function $f(\Delta V_{TH1}, \Delta V_{TH3})$ in this example, as shown in Fig. 13. Hence, when G-S applies binary search in Algorithm 3 to the random variable α_1 or α_2 that defines the orientation, it may not find a 1-D interval that fully covers the failure region. In this case, the Gibbs sample is drawn from a subset of the failure region, instead of the actual failure region. However, it is likely that the binary search converges to a different 1-D interval during the next iteration step. It, in turn, helps to move the Gibbs sample over a long distance. For this reason, the random samples iteratively generated by G-S can be widely distributed over the high-probability failure region eventually.

VI. Conclusion

In this paper, a novel Gibbs sampling method was proposed for efficient failure rate prediction of SRAM circuits. The proposed Gibbs sampling algorithm was implemented for both Cartesian and spherical coordinate systems. It adaptively explored the variation space so that a large number of random samples fell into the failure region and the SRAM failure probability could be accurately estimated with a low computational cost. In particular, it iteratively sampled a sequence of 1-D PDFs by an efficient inverse-transform method. As was demonstrated by our experimental results for a 90 nm SRAM cell, the proposed Gibbs sampling achieved 1.4∼4.9× runtime speedup over other state-of-the-art techniques, when a high prediction accuracy was required (e.g., the relative error defined by the 99% confidence interval reached 5%). In addition, we further demonstrated an important example for which the proposed Gibbs sampling algorithm accurately estimated the correct failure probability while the traditional techniques failed to work. The Gibbs sampling technique can be further incorporated into a statistical optimization environment for accurate and efficient parametric yield optimization of SRAM circuits.

On the other hand, it is important to mention that the proposed Gibbs sampling method was particularly developed to handle a single continuous failure region only. In addition, the proposed Gibbs sampling technique can be computationally inefficient for high-dimensional problems where there are a large number of random variables in (1) (e.g., $M \geq 30$). In these high-dimensional applications, Gibbs sampling only samples one random variable at each iteration step, thereby resulting in slow convergence. Finally, for a given circuit where the failure region is unknown, it remains an open question how to automatically select the appropriate importance sampling algorithm (e.g., MIS, MNIS, G-C, or G-S) to achieve high estimation accuracy and low computational cost. These issues will be further studied in our future research.

Appendix
Proof of Theorem 1

It has been shown in [17] that if the random variables $\{\alpha_m; m = 1, 2, \ldots, M\}$ are mutually independent and standard Normal, the variables $\{\alpha_m/||\boldsymbol{\alpha}||_2;\ m = 1, 2, \ldots, M\}$ are uniformly distributed on the surface of a sphere with unit radius. Hence, for a given value of r, the random variables $\{x_m; m = 1, 2, \ldots, M\}$ defined in (11) are uniformly distributed on the surface of a sphere $||\mathbf{x}||_2 = r$. In other words, for all points on the surface of the sphere $||\mathbf{x}||_2 = r$, their PDF values are identical. In order to calculate the joint PDF for $\mathbf{x}$, we consider the sphere $||\mathbf{x}||_2 = r$ and add a small perturbation Δr to the radius, as shown by the 3-D example in Fig. 15. If the perturbation Δr is sufficiently small (i.e., $\Delta r \to 0$), the values of $f(\mathbf{x})$ at all points within the region $r \leq ||\mathbf{x}||_2 \leq r + \Delta r$ are identical.

As shown in (13), the radius r follows the Chi distribution with M degrees of freedom. The CDF of r is represented as follows [20]:

$$F(r) = \frac{2}{\Gamma(0.5 \cdot M)} \cdot \left(\frac{1}{2}\right)^{0.5 \cdot M} \cdot \int_0^r t^{M-1} \cdot e^{-0.5 \cdot t^2} \cdot dt. \quad (34)$$

Hence, the probability for $r \leq ||\mathbf{x}||_2 \leq r + \Delta r$ where $\Delta r \to 0$ is equal to

$$\begin{aligned} \Delta F &= F(r + \Delta r) - F(r) = \frac{dF}{dr} \cdot \Delta r \\ &= \frac{2}{\Gamma(0.5 \cdot M)} \cdot \left(\frac{1}{2}\right)^{0.5 \cdot M} \cdot r^{M-1} \cdot e^{-0.5 \cdot r^2} \cdot \Delta r. \end{aligned} \quad (35)$$

On the other hand, the volume of an M-dimensional sphere can be written as follows [21]:

$$V(r) = \frac{2 \cdot \pi^{0.5 \cdot M}}{M \cdot \Gamma(0.5 \cdot M)} \cdot r^M. \quad (36)$$

Hence, the volume for the region $r \leq ||\mathbf{x}||_2 \leq r + \Delta r$ where $\Delta r \rightarrow 0$ is equal to

$$\begin{aligned} \Delta V &= V(r + \Delta r) - V(r) = \frac{dV}{dr} \cdot \Delta r \\ &= \frac{2}{\Gamma(0.5 \cdot M)} \cdot \pi^{0.5 \cdot M} \cdot r^{M-1} \cdot \Delta r. \end{aligned} \tag{37}$$

Since the values of $f(\mathbf{x})$ at all points within the region $r \leq ||\mathbf{x}||_2 \leq r + \Delta r$ ($\Delta r \rightarrow 0$) are identical, $f(\mathbf{x})$ is equal to the ratio between ΔF in (35) and ΔV in (37)

$$f(x) = \frac{\Delta F}{\Delta V} = \left(\frac{1}{2 \cdot \pi} \right)^{0.5 \cdot M} \cdot e^{-0.5 \cdot r^2}. \tag{38}$$

Substituting (12) into (38) yields

$$f(x) = \left(\frac{1}{2 \cdot \pi} \right)^{0.5 \cdot M} \cdot \exp\left(-\frac{x_1^2 + x_2^2 + \cdots + x_M^2}{2} \right). \tag{39}$$

The PDF in (39) is exactly equal to that of the multivariate Normal distribution, as shown in (1).

REFERENCES

[1] C. Dong and X. Li, "Efficient SRAM failure rate prediction via Gibbs sampling," in *Proc. Des. Automat. Conf.*, 2011, pp. 200–205.

[2] A. Bhavnagarwala, X. Tang, and J. Meindl, "The impact of intrinsic device fluctuations on CMOS SRAM cell stability," *IEEE J. Solid-State Circuits*, vol. 36, no. 4, pp. 658–665, Apr. 2001.

[3] R. Heald and P. Wang, "Variability in sub-100 nm SRAM designs," in *Proc. Int. Conf. Comput.-Aided Des.*, 2004, pp. 347–352.

[4] B. Calhoun, Y. Cao, X. Li, K. Mai, L. Pileggi, R. Rutenbar, and K. Shepard, "Digital circuit design challenges and opportunities in the era of nanoscale CMOS," *Proc. IEEE*, vol. 96, no. 2, pp. 343–365, Feb. 2008.

[5] S. Mukhopadhyay, H. Mahmoodi, and K. Roy, "Statistical design and optimization of SRAM cell for yield enhancement," in *Proc. Int. Conf. Comput.-Aided Des.*, 2004, pp. 10–13.

[6] K. Agarwal and S. Nassif, "Statistical analysis of SRAM cell stability," in *Proc. Des. Automat. Conf.*, 2006, pp. 57–62.

[7] M. Abu-Rahma, K. Chowdhury, J. Wang, Z. Chen, S. Yoon, and M. Anis, "A methodology for statistical estimation of read access yield in SRAMs," in *Proc. Des. Automat. Conf.*, 2008, pp. 205–210.

[8] R. Kanj, R. Joshi, and S. Nassif, "Mixture importance sampling and its application to the analysis of SRAM designs in the presence of rare failure events," in *Proc. Des. Automat. Conf.*, 2006, pp. 69–72.

[9] A. Singhee and R. Rutenbar, "Statistical blockade: A novel method for very fast Monte Carlo simulation of rare circuit events, and its application," in *Proc. Des. Automat. Test Eur.*, 2007, pp. 1–6.

[10] L. Dolecek, M. Qazi, D. Shah, and A. Chandrakasan, "Breaking the simulation barrier: SRAM evaluation through norm minimization," in *Proc. Int. Conf. Comput.-Aided Des.*, 2008, pp. 322–329.

[11] J. Wang, S. Yaldiz, X. Li, and L. Pileggi, "SRAM parametric failure analysis," in *Proc. Des. Automat. Conf.*, 2009, pp. 496–501.

[12] J. Jaffari and M. Anis, "Adaptive sampling for efficient failure probability analysis of SRAM cells," in *Proc. Int. Conf. Comput.-Aided Des.*, 2009, pp. 623–630.

[13] K. Katayama, S. Hagiwara, H. Tsutsui, H. Ochi, and T. Sato, "Sequential importance sampling for low-probability and high-dimensional SRAM yield analysis," in *Proc. Int. Conf. Comput.-Aided Des.*, 2010, pp. 703–708.

[14] M. Qazi, M. Tikekar, L. Dolecek, D. Shah, and A. Chandrakasan, "Loop flattening and spherical sampling: Highly efficient model reduction techniques for SRAM yield analysis," in *Proc. Des. Automat. Test Eur.*, 2010, pp. 801–806.

[15] C. Gu and J. Roychowdhury, "An efficient, fully nonlinear, variability-aware non-Monte-Carlo yield estimation procedure with applications to SRAM cells and ring oscillators," in *Proc. Asia South Pacific Des. Automat. Conf.*, 2008, pp. 754–761.

[16] C. Andrieu, N. Freitas, A. Doucet, and M. Jordan, "An introduction to MCMC for machine learning," *Mach. Learning*, vol. 50, no. 1, pp. 5–43, Jan. 2003.

[17] G. Marsaglia, "Choosing a point from the surface of a sphere," *Ann. Math. Statist.*, vol. 43, no. 2, pp. 645–646, Apr. 1972.

[18] H. Zhang, T. Chen, M. Ting, and X. Li, "Efficient design-specific worst-case corner extraction for integrated circuits," in *Proc. Des. Automat. Conf.*, 2009, pp. 386–389.

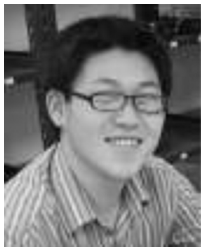
[19] A. Chandrakasan, W. Bowhill, and F. Fox, *Design of High-Performance Microprocessor Circuits*. New York: Wiley-IEEE Press, 2000.

[20] A. Papoulis and S. Pillai, *Probability, Random Variables and Stochastic Processes*. New York: McGraw-Hill, 2001.

[21] M. Kendall, *A Course in the Geometry of n Dimensions*. New York: Dover, 2004.

[22] G. Fishman, *A First Course in Monte Carlo*. Pacific Grove, CA: Duxbury Press, Oct. 2005.

[23] C. Bishop, *Pattern Recognition and Machine Learning*. Englewood Cliffs, NJ: Prentice-Hall, 2007.



Shupeng Sun (S'11) received the B.S. degree in automation from Tsinghua University, Beijing, China, in 2010. He is currently pursuing the Ph.D. degree in electrical and computer engineering with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA.

His current research interests include statistical design and optimization of low-power SRAM circuits.

Yamei Feng received the B.S. degree in information science and electronic engineering from Zhejiang University, Hangzhou, China, in 2007, and the M.S. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, in 2011.

She is currently with the Department of Electrical and Computer Engineering, Carnegie Mellon University. Her current research interests include computer-aided design and signal processing.

Changdao Dong received the B.S. degree in computer science from Tsinghua University, Beijing, China, in 2007, and the M.S. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, in 2011.

He is currently with the Department of Electrical and Computer Engineering, Carnegie Mellon University. He was a Software Development Intern with IBM CSDL, Beijing, in 2009. His current research interests include statistical analysis and modeling for memory circuits under process variations.

Xin Li (S'01–M'06–SM'10) received the B.S. and M.S. degrees in electronics engineering from Fudan University, Shanghai, China, in 1998 and 2001, respectively, and the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, in 2005.

He is currently an Assistant Professor with the Department of Electrical and Computer Engineering, Carnegie Mellon University. His current research interests include computer-aided design, neural signal processing, and power system analysis and design.

Dr. Li has been an Associate Editor of the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS since 2012. He has served on the technical program committees of DAC and ICCAD. He is a recipient of the NSF CAREER Award in 2012, the DAC Best Paper Award in 2010, and two ICCAD Best Paper Awards in 2004 and 2011.