UCLA

UCLA Electronic Theses and Dissertations

Title

Stochastic Yield Analysis of Rare Failure Events in High-Dimensional Variation Space

Permalink

https://escholarship.org/uc/item/58f61961

Author

Shi, Xiao

Publication Date

2020

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

UNIVERSITY OF CALIFORNIA

Los Angeles

Stochastic Yield Analysis of Rare Failure Events in High-Dimensional Variation Space

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Electrical Engineering

by

Xiao Shi

2020

© Copyright by

Xiao Shi

2020

ABSTRACT OF THE DISSERTATION

Stochastic Yield Analysis of Rare Failure Events in
High-Dimensional Variation Space

by

Xiao Shi
Doctor of Philosophy in Electrical Engineering
University of California, Los Angeles, 2020
Professor Lei He, Chair

As semiconductor industry kept shrinking the feature size to nanometer scale, circuit reliability has become an area of growing concern due to the uncertainty introduced by process variations. For highly-replicated standard cells, the failure event for each individual component must be extremely rare in order to maintain sufficiently high yield rate. Existing yield analysis approaches works fine at low dimension, but less effective either when there are a large amount of circuit parameters, or when the failure samples are distributed in multiple regions. In this thesis, four novel high sigma analysis approaches have been proposed.

First, we propose an adaptive importance sampling (AIS) algorithm. AIS has several iterations of sampling region adjustments, while existing methods pre-decide a static sampling distribution. At each iteration, AIS generates samples from current proposed distribution. Next, AIS carefully assigns weight to each sample based on its tilted occurrence probability between failure region and current failure region distribution. Then we design two adaptive frameworks based on Resampling and population Metropolis-Hastings (MH) to iteratively search for failure regions.

Second, we develop an Adaptive Clustering and Sampling (ACS) method to estimate the failure rate of high-dimensional and multi-failure-region circuit cases. The basic idea of the algorithm is to cluster failure samples and build global sampling distribution at each iteration. Specifically, in clustering step, we propose a multi-cone clustering method, which partitions the parametric space and clusters failure samples. Then global sampling distribution is constructed from a set of weighted Gaussian distributions. Next, we calculate importance weight for each sample based on the discrepancy between sampling distribution and target distribution. Failure probability is updated at the end of each iteration. This clustering and sampling procedure proceeds iteratively until all the failure regions are covered.

Moreover, two meta-model based approaches are proposed for high sigma analysis. The Low-Rank Tensor Approximation (LRTA) formulate the meta-model in tensor space by representing a multi-way tensor into a finite sum of rank-one tensor. The polynomial degree of our LRTA model grows linearly with circuit dimension, which makes it especially promising for high-dimensional circuit problems. Then we solve our LRTA model efficiently with a robust greedy algorithm, and calibrate iteratively with an adaptive sampling method. The meta-model based importance sampling (MIS) method utilizes Gaussian Process meta-model to construct quasi-optimal importance sampling distribution, and performs Markov Chain Monte Carlo (MCMC) simulation to generate new samples from the proposed distribution. By updating our global Importance Sampling estimator in an iterated framework, MIS leads to better efficiency and higher accuracy than traditional importance sampling methods. Experiment results validate that the proposed approaches are 3 orders faster than Monte Carlo, and more accurate than both academia solutions such as importance sampling and classification based methods, and industrial solutions such as mixture IS used by Intel.

The dissertation of Xiao Shi is approved.

Puneet Gupta

Sudhakar Pamarti

Yong Chen

Lei He, Committee Chair

University of California, Los Angeles

2020

To My Lovely Family and Friends.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

VITA

2008–2012	B.S., Department of Electrical Engineering, Southeast University, Nanjing, China.
2013–2015	M.S., Department of Electrical and Computer Engineering, University of California, Los Angeles, California, USA.
2015–present	Ph.D. program, Department of Electrical and Computer Engineering, University of California, Los Angeles, California, USA. – Teaching Assistant, Modeling of VLSI Circuits and Systems, – Teaching Assistant, Circuit Measurement Laboratory, – Graduate Student Researcher, UCLA Design Automation Lab, 2013–present.

PUBLICATIONS

Shi, X., Yan, H., Wang, J., Zhang, J., Shi, L., He, L. (2020). An Efficient Adaptive Importance Sampling Method for SRAM and Analog Yield Analysis. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems.*

Shi, X. , Yan, H. , Zhang, J. , Huang, Q. , Shi, L. and He, L. , "Efficient Yield Analysis for SRAM and Analog Circuits using Meta-Model based Importance Sampling Method," In *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), Westminster, CO, USA, 2019,* pp. 1-8.

Shi, X., Yan, H., Huang, Q., Zhang, J., Shi, L., He, L. (2019, June). Meta-Model based

High-Dimensional Yield Analysis using Low-Rank Tensor Approximation. In *Proceedings of the 56th Annual Design Automation Conference 2019* (pp. 1-6).

Shi, X., Yan, H., Wang, J., Xu, X., Liu, F., Shi, L., He, L. (2019, April). Adaptive Clustering and Sampling for High-Dimensional and Multi-Failure-Region SRAM Yield Analysis. In *Proceedings of the 2019 International Symposium on Physical Design* (pp. 139-146).

Shi, X., Liu, F., Yang, J., He, L. (2018, June). A fast and robust failure analysis of memory circuits using adaptive importance sampling method. In *2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)* (pp. 1-6). IEEE.

CHAPTER 1

Introduction

1.1 Background

The scale of semiconductor industry has been migrating to the end of Moore's Law in the past few decades. The leading Foundries TSMC and Samsung have announced their 5-nanometer manufacturing process [Moo19], respectively. Manufacturer favors the latest technology because it is associated with emerging applications such as high-end mobile and high-performance computing. TSMC says that its 5-nm process offers a 15 percent speed gain or a 30 percent improvement in power efficiency. Samsung claims a 10 percent performance improvement or a 20 percent efficiency improvement. However, in modern IC manufacturing, it has become increasingly challenging to maintain high-precision and high-reliability.

Circuit variations can be categorized into two groups. [GP08] On the one hand, "process variations" or "static variations" include the variations occur during the manufacturing process (e.g. mask misalignment, stepper focus, etc). On the other hand, "environmental variations" or "dynamic variations" denote those occur during circuit operation that vary over time (e.g. temperature, power supply voltage, etc). In this thesis, we only focus on process variations, which play a dominating role among the sources that introducing uncertainty to circuit behavior [EBSLM97, CCS04].

Moreover, process variations have become a more important issue as semiconductor technology scales down. The performance characteristics of a MOSFET device is defined by the physical and electrical parameters, including threshold voltage (Vth), channel length (L), channel width (W) and oxide thickness (Tox), etc. It is straightforward to see that large variations on these parameters can be converted into large amount of variations in circuit behavior(e.g., leakage power, timing delay, output swing, etc.), and thereby more likely to fail various circuit performance specifications.

1.2 Motivation

In order to analyze the impact of process variations, automated tools are required at pre-silicon phase. As semiconductor technology scales down to smaller feature size, process variation effects resulting in circuits targeting a lower yield, which is more susceptible to reliability issues. For example, an SRAM array typically contains millions of replicated components such as SRAM bit cell, sense amplifier and peripheral circuit, which are designed to be exactly the same circuit parameters. Since a single failure can cause catastrophic failure of the entire SRAM array, each replicated component must be extremely robust under process variations so that the overall yield rate of SRAM array can meet the design specification. Generally, the requirement of SRAM bit cell failure rate need to be less than 10^{-8} to 10^{-6} (6-7 sigma). However, traditional deterministic methods such as [SR09, Li10] cannot solve problems with this dimension. Therefore, it is urgent for circuit designers to sought to statistical yield analysis tools.

To enable circuit designers to sketch the stochastic behavior of a complex system under the impact of process variations, many existing methodologies have been proposed in the past few years [Nas01, LLGP04, DQSC08, VWG06, GYH11]. In general, previous works are based on the assumption that target design is linear circuit or the target specification follows the Gaussian distribution. However, the major challenge for existing methods has two aspects:

- Strong non-linearity: the circuit responses of modern custom circuit blocks have strong nonlinear relationships with input parameters. Therefore, traditional first-order and second-order reliability methods are not applicable. It is extremely challenging to accurately model the stochastic behavior of custom circuits.

- High dimensionality: the target circuit cases for existing yield analysis methods are memory standard cells, which are relatively small-scale. However, there is increasing demand from market to perform yield analysis on various analog circuits, such as amplifiers and comparators.

1.3 Organization of the Dissertation

The research presented in this dissertation mainly focuses on process variation modeling and analysis using numerical and statistical techniques. The remaining parts of this dissertation are organized as follows:

- **Chapter 2: Adaptive Importance Sampling for Fast Yield Analysis**

  An improved importance sampling method is presented for the failure analysis of memory circuits where circuit failure is a rare event.

- **Chapter 3: Adaptive Clustering and Sampling for Fast Yield Analysis**

  Adaptive Clustering and Sampling (ACS) explicitly uses spherical clustering algorithm to locate the clusters of failure samples and considers those clusters as failure regions. Moreover, it also modifies mixture importance sampling to apply the mean-shift technique to multiple failure regions.

- **Chapter 4: Meta-Model based High-Dimensional Yield Analysis using Low-Rank Tensor Approximation**

  A novel and efficient polynomial-based meta-model with tensor structure is presented in this chapter to tackle the challenging high-dimensional yield analysis problem.

- **Chapter 5: Meta-Model based Importance Sampling Algorithm**

  A novel Gaussian Process meta-model based approach is proposed in this chapter to build up quasi-optimal importance sampling density.

- **Chapter 6: Conclusion**

  The conclusion and future works are discussed.

CHAPTER 2

Adaptive Importance Sampling for Fast Yield Analysis

2.1 Introduction

As microelectronic devices shrink to nano-meter scale, process variation has become a growing concern for efficient circuit sizing and design. For highly-replicated memory circuits or analog modules, the tolerable failure probability is extremely small. In order to achieve a robust design, we need to establish very accurate estimation of failure probability for each circuit component. Conventional circuit simulation approaches perform worst case analysis (WCA) or build analytical model to deterministically estimate the failure probability. It is, however, infeasible in the rare-event scenario.

In general, the probability estimation of rare event is achieved by sampling methods. Among those methods, Monte Carlo (MC) approach remains the gold standard, which repeatedly generates samples and evaluates circuit performance with transistor-level SPICE simulation. However, MC is extremely time-consuming under the rare-event scenario because millions of simulations are required to capture one single failure event.

Prior Works. In order to avoid expensive MC simulations, more efficient approaches have been proposed to sample from the likely-to-fail regions. We can group these methods into three major categories:

(1) Classification: Boundary Searching (BS) method in this category has been developed to speed up the failure probability estimation through constructing the whole boundary of the failure region. Then failure probability can be estimated by computing the hyper-volume of the failure region without extra simulations. For example, Statistical Blockade (SB) [SR07] applies a classifier to filter the more likely-to-fail MC samples and only simulate these samples. A safety margin is applied in [SR07] to decrease the classification error. More recently, Recursive SB [SWCR08] and RE-scope [WXK+14] construct conditional classifier and SVM-based nonlinear classifier, respectively. However, training such efficient classifiers is expensive in high dimension and the complexity grows exponentially as failure probability decreases.

(2) Meta-modeling: Modeling methods are employed in recent works to build circuit evaluators in replacement of circuit simulators. It maps the variation parameters into the circuit metrics. Evaluating meta-models is much faster than running circuit simulators, which drastically reduces computational complexity during yield estimation. For example, the approaches in [WLY+18, YYW14] build surrogate models based on Gaussian process regression and radial basis function network, respectively. Moreover, the approach in [SYH+19] utilizes low-rank tensor approximation (LRTA) method to train an efficient meta-model. However, the estimation accuracy is highly sensitive to the accuracy of the meta-model, which generally requires massive number of training samples.

(3) Importance Sampling: Importance Sampling (IS) has been proposed based on the insight that drawing samples in the likely-to-fail regions with a distorted sampling distribution can improve the estimation accuracy and meanwhile accelerate the estimation convergence. The choice of distorted distributions has a significant impact on the performance of IS. For example, Mixture Importance Sampling (Mix-IS) [KJN06] mixes three distributions: a uniform distribution, the original distribution and a shifted distribution centered at the failure region. The approaches in [DQSC08], [QTD+10] simply shift the center of sampling distribution toward the minimum L2-norm point of a set of failure samples, while the method in [WGCH14] shifts the sample mean to the centroid of failure samples. In order to tackle multi-failure-region problems, approaches in [WBH16], [WYL+16] attempt to construct multiple mean shift vectors to obtain full coverage of different failure regions. However, all these methods construct a static likely-to-fail region to sample from, which makes them vulnerable to poor initializations.

In this chapter, we present an efficient and accurate adaptive importance sampling (AIS) method for the failure probability estimation of various circuit cases. At each iteration, AIS generates samples from a family of partial distributions. Next, AIS carefully assigns weight to each sample according to its tilted probability density between target distribution and current sampling distribution. Then AIS updates sampling distribution to prevent weight degeneracy, which tends to discard low-weight samples and regenerate high-weight samples. Unlike conventional static importance sampling approaches, this procedure can proceed iteratively to search for failure regions.

Although the idea of AIS was initially developed in the statistics field, it was previously unknown how to adapt AIS to the circuit failure probability estimation problems. In general, our major contribution is presented in three aspects: first, a primary difficulty with AIS algorithm is to initialize the parameterized sampling distribution. To solve this issue, hyperspherical presampling is performed in order to collect a number of failure events and sketch initial sampling distribution. Second, we select the circuit failure event distribution as the AIS target distribution, which bridges the gap between AIS algorithm and circuit yield analysis problem. Moreover, we develop two distinct sampling distribution adaptation methods, including Resampling method to sample with replication from partial distributions, and population Metropolis-Hastings (MH) to iteratively replace distribution with accept/reject criteria. To the best of our knowledge, this is the first work on successfully developing adaptive importance sampling method for circuit failure probability estimation.

2.2 Background

2.2.1 Rare Event Analysis

Let $f(X)$ denote multivariate probability density function (PDF) of circuit process variation X. Let Y be the performance merit of interest, such as memory read/write time, amplifier gain, etc. Y usually requires expensive transistor-level circuit simulation to evaluate.

In general, it is of great interest to estimate the probability that Y belongs to a subset S of the entire parametric space. For example, under circuit failure rate estimation scenario, a tiny subset S can denote the “failure region” which includes all the failed samples. Thereby, we introduce indicator function $I(X)$ to identify pass/fail of Y:

$$I(X) = \begin{cases} 0, & if \quad Y \notin S \\ 1, & if \quad Y \in S \end{cases} \tag{2.1}$$

Then the probability P_{fail} can be calculated as:

$$P_{fail} = P(Y \in S) = \int I(X) \cdot f(X) dX \tag{2.2}$$

Note that the integral in Equation (2.2) is intractable because $I(X)$ is unavailable in analytical form. Conventionally, MC method enumerates a sample set $\{x_i\}_{i=1}^{N}$ according to $f(X)$ and evaluates their indicator function values to generate an unbiased estimation $\hat{P}_{fail}$:

$$\hat{P}_{fail} = \hat{P}(Y \in S) = \frac{1}{N} \sum_{i=1}^{N} I(x_i) \xrightarrow{N \to +\infty} P(Y \in S) \tag{2.3}$$

2.2.2 Importance Sampling

When S denotes a rare event, such as circuit failure, standard MC becomes infeasible because it requires millions of simulations to capture one single failure event. To avoid massive MC simulations, IS has been introduced to sample from a "distorted" sampling distribution $g(X)$ that tilts towards the failure region S.

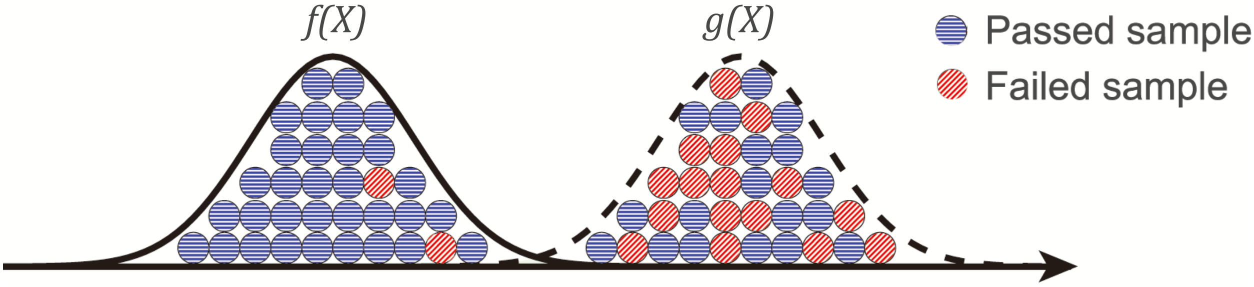


Figure 2.1: 1D illustrative plot of mean-shift importance sampling. Failed samples are more likely to be captured with appropriate shift vector.

As shown in one-dimensional illustrative Figure 2.1 , $g(X)$ contains more statistically likely-to-fail samples. Here failure probability P_{fail} can be expressed as:

$$P_{fail} = P(Y \in S) = \int I(X) \cdot \frac{f(X)}{g(X)} \cdot g(X) dX \tag{2.4}$$

$$= \int I(X) \cdot w(X) \cdot g(X) dX \tag{2.5}$$

where $w(X)$ denotes the importance weight between original PDF $f(X)$ and shifted $g(X)$. $w(X)$ compensates for the discrepancy between $f(X)$ and $g(X)$. Then the unbiased IS estimator $\hat{P}_{IS,fail}$ can straightforwardly collect samples $\{x_j\}_{j=1}^{M}$ from $g(X)$:

$$\hat{P}_{IS,fail} = \hat{P}(Y \in S) = \frac{1}{M} \sum_{i=1}^{M} w(x_j) I(x_j) \xrightarrow{M \to +\infty} P(Y \in S) \tag{2.6}$$

We notice that IS sample set $\{x_j\}_{j=1}^{M}$ in Equation (2.6) has much smaller size than MC sample set $\{x_i\}_{i=1}^{N}$ in Equation (2.3) because failure in $g(X)$ is not a rare event. Theoretically, the optimal sampling distribution $g^{opt}(X)$ is the failure event distribution $\pi(X)$:

$$g^{opt}(X) = \pi(X) = \frac{I(X) \cdot f(X)}{P_{fail}} \tag{2.7}$$

However, $g^{opt}(X)$ cannot be evaluated with Equation (2.7) directly because the analytical expression of $I(X)$ is unknown and P_{fail} is the target failure probability to be estimated. In practice, most existing approaches attempt to shift the sample mean toward either pass/fail boundary [DQSC08], centroid of a cluster of failure samples [WGCH14], or centroids of multiple clusters [WBH16]. However, current mean-shift IS implementations suffer from two major drawbacks:

First, existing IS approaches consist of two stages: a presampling stage and an importance sampling stage. Generally speaking, presampling is to construct initial sampling distribution and importance sampling is to evaluate P_{fail}. However, these algorithms are vulnerable to poor initial conditions. It is extremely time-consuming to construct a desired shift vector in the Presampling stage.

Moreover, the importance weight $w(X)$ is substantially biased because the huge discrepancy between $f(X)$ and $g(X)$. In this way, only a few samples contribute to the failure probability calculation. It is the main challenge to obtain consistent estimation.

2.3 Adaptive Importance Sampling Algorithm

In this section, we present our Adaptive Importance Sampling (AIS) approach. It adaptively improves the sampling distribution by searching for failure regions through an adaptive sampling strategy.

2.3.1 Algorithm Description

The objective of AIS is to approximate the target failure event distribution, $\pi(x)$, by constructing a mixture of partial distributions. Since the analytical expression of $\pi(x)$ is unavailable, we cannot directly draw samples from it. Our novel approach is to "learn" from the past simulations and construct a sequence of adaptive sampling distributions.

The pseudo-code of AIS algorithm is summarized in Algorithm 1. First of all, AIS starts with N initial sample points $\{\mu_i^{(0)}\}_{i=1}^N$. The first sampling distribution is defined as a mixture density $g^{(0)}(x)$ built from N partial Gaussian distributions $q_i^{(0)}(x)$ with location parameters $\mu_i^{(0)}$ and covariance matrices Σ_i. At each iteration t, we independently draw N samples $x_i^{(t)}$ from the N partial distributions $q_i^{(t-1)}(x)$. Next, we compute the importance weight $w_{i,t}$ of each sample $x_i^{(t)}$ by evaluating the ratio of target density $\pi(x)$ and mixture density $g^{(t-1)}(x)$. Thus our failure probability estimator can be expressed as the average of all the importance weights. Then we update the location parameters $\mu_i^{(t)}$ based on normalized samples weights $\{\bar{w}_{i,t}\}_{i=1}^N$ through a carefully designed sampling distribution adaptation process. It can balance sample weights and thus prevent weight degeneracy. This sampling and adaption procedure can proceed iteratively until the discrepancy between the target distribution and sampling distribution is not reduced anymore.

2.3.2 Parameter Tuning

One major challenge in implementing our AIS algorithm is that the selection of parameters can significantly affect the efficiency of our iterative search procedure. Properly chosen parameters can lead to higher sample diversity, faster convergence, and better capability

Algorithm 1: AIS Algorithm

```
Initialization:
1. Set iteration index $t = 0$, construct N partial Gaussian distributions
    $q_i^{(0)}(x) = q_i^{(0)}(x|\mu_i^{(0)}, \Sigma_i) \quad i = 1, \cdots, N$
where $\mu_i$ is the location parameters and $\Sigma_i$ is the covariance matrices.
2. Set initial sampling distribution
    $g^{(0)}(x) = \frac{1}{N} \sum_{i=1}^{N} q_i^{(0)}(x)$
repeat
    Update iteration index $t = t + 1$ .
    1. Sample Propagation:
    Draw a new set of N samples $\{x_i^{(t)}\}_{i=1}^{N}$ in the parametric space:
        $x_i^{(t)} \sim q_i^{(t-1)}(x|\mu_i^{(t-1)}, \Sigma_i) \quad i = 1, \cdots, N$
    2. Weighting:
    Compute incremental importance weights:
        $w_{i,t} = \frac{\pi(x)}{g^{(t-1)}(x)} = \frac{f(x)I(x)}{\frac{1}{N}\sum_{i=1}^{N} q_i^{(t-1)}(x)} \quad i = 1, \ldots, N$
    3. Estimation:
    Update the unbiased estimator:
        $\hat{P}_{fail,t} = \frac{1}{tN} \sum_{j=1}^{t} \sum_{i=1}^{N} w_{i,t} \quad i = 1, \ldots, N$
    4. Normalization:
    Normalize importance weights:
        $\bar{w}_{i,t} = \frac{w_{i,t}}{\sum_{i=1}^{N} w_{i,t}} \quad i = 1, \cdots, N$
    5. Sampling Distribution Adaptation: Update location parameters
     $\{\mu_i^{(t-1)}\}_{i=1}^{N}$ based on
    normalized weight $\{\bar{w}_{i,t-1}\}_{i=1}^{N}$, and generate improved sampling distribution
     $g^{(t)}(x)$
until Relative standard deviation (FOM): $\rho = \frac{\sqrt{\sigma^2_{\hat{P}_{fail}}}}{\hat{P}_{fail}}$;
```

to explore parametric space. In this section, we mainly discuss initial location parameters, partial distributions and covariance matrices.

2.3.2.1 Initial Location Parameters $\mu_i^{(0)}$

As our AIS algorithm is driven by its adaptive mechanism to explore failure regions, it is worthwhile to develop a good initial experimental design. Here we propose a hyperspherical presampling method to capture more failure samples with minimum number of simulations. The main concept is to collect failure samples distributed on multiple hyperspherical surfaces, and use them as initial location parameters.

As shown in Figure 2.2, we start with original distribution, which denotes a unit hypersphere in parametric space. We notice that most likely no failure event is captured under original distribution. To cover multiple failure regions, we gradually increase the radius of hypersphere until N failure samples are captured. Here variable N, which denotes the number of partial distributions in Algorithm 1 can be arbitrarily user-defined. The trade-off between estimation accuracy and algorithm complexity is obvious: we can achieve higher estimation accuracy as N increases at the expense of longer simulation runtime. In our experiments, we choose N to be 128 and collect 1000 samples on each hypersphere.

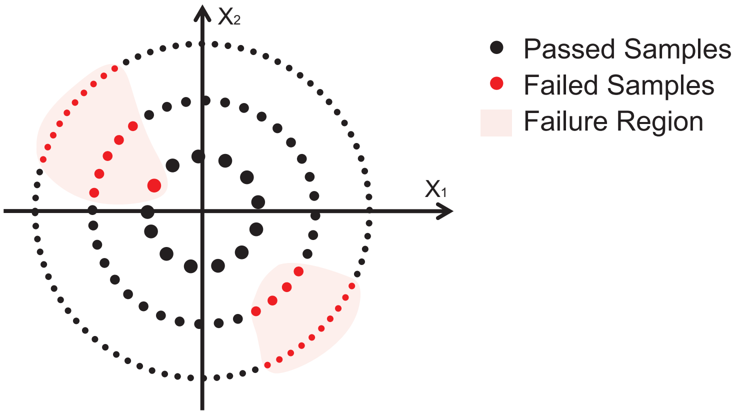


Figure 2.2: Hyperspherical presampling. Samples on outer spherical surfaces are smaller dots, denoting smaller probability to be captured.

An alternative view of our hyperspherical presampling method is to project high dimensional samples onto surface of multiple hyperespheres, which is a dimension reduction process. This process is capable of accelerating the exploration in entire open space while maintaining the sample diversity.

2.3.2.2 Partial Distribution $q_i^{(t)}(x)$

In each iteration of our AIS algorithm, sampling distribution $g^{(t)}(x)$ is constructed by averaging out the partial distribution $q_i^{(t)}(x)$ to approximate the target failure event distribution $\pi(x)$. This concept is equivalent to the methodology in Kernel Density Estimation (KDE), where each $q_i^{(t)}(x)$ represents the kernel function. For different applications, there exists various parameterized distributions as kernel functions, such as Gaussian distributions, Beta distributions [Che99] and Boltzmann distributions [DCB$^+$10]. According to the Gaussian nature of circuit variables, we place a multivariate Gaussian distribution centered at each of the location parameters, which indicates a probability mass in the parametric space.

2.3.2.3 Covariance Matrices Σ_i

Another parameter can be tuned is the covariance matrices Σ_i of aforementioned partial distribution $q_i^{(t)}(x)$. In practice, due to weak correlation between different circuit variables, this covariance matrices is diagonal or can be jointly approximate diagonalized. For matrices entries with larger values, corresponding circuit dimensions tend to present higher freedom to explore failure regions. Matrices entries with smaller values, on the contrary, concentrate sample points and achieve faster convergence. In our AIS implementation, our covariance matrices are not adapted between iterations, but values change among different entries to control the freedom of different circuit dimensions.

2.4 Sampling Distribution Adaptation

As explained in Section II, standard IS suffers from weight degeneracy during the estimation. Due to the mismatch between sampling distribution and target distribution, a few samples may have extremely large weight values while others are trivial. In this case, the failure probability estimation is highly dependent on a small sample set, which makes the estimator unstable and fail to converge.

In this section, we propose two different sampling strategies to update the sampling distribution. The first Resampling method samples with replication from the partial distributions. Distributions with larger weights tend to generate more samples in the next iteration. The second population Metropolis-Hastings (MH) method updates partial distributions through an accept/reject rule. Then we compare these two methods in terms of simulation cost and failure region coverage.

2.4.1 Resampling

Our first method is driven by multinomial resampling technique to iteratively improve partial distributions. The main idea is to replicate distributions with larger weights and eliminate those with smaller weights.

Figure 2.3 is an illustrative view for certain iteration t. It demonstrates the adjustment of normalized weight $\{\bar{w}_{i,t}\}_{i=1}^{N}$ and sampling distribution $g^{(t)}(x)$ after resampling for N times. The specific steps are as follows. After sample set $\{x_i^{(t)}\}_{i=1}^{N}$ is generated from Sample Propagation step, we first locate the partial distributions $\{\mu_i^{(t)}\}_{i=1}^{N}$ centered at $\{x_i^{(t)}\}_{i=1}^{N}$. Then we resample $\{\mu_i^{(t)}\}_{i=1}^{N}$ with replication according to their normalized weights $\{\bar{w}_{i,t}\}_{i=1}^{N}$. Thus we can obtain new sampling distribution $g^{(t)}(x)$ by averaging out current partial distributions $\{q_i^{(t)}(x|\mu_i^{(t)}, \Sigma_i)\}_{i=1}^{N}$. In the sample generation step of iteration $(t+1)$, a set of updated random measure will populate in the region of large weights. In this way, this resampling approach effectively mitigates weight degeneracy, and guarantees each sample uniformly contribute to the estimator.

2.4.2 Population Metropolis-Hastings Sampler

Our second method updates the sampling distribution from a different perspective, which is a population MH sampler. The intuitive idea of population MH is to repeatedly replace a “bad”

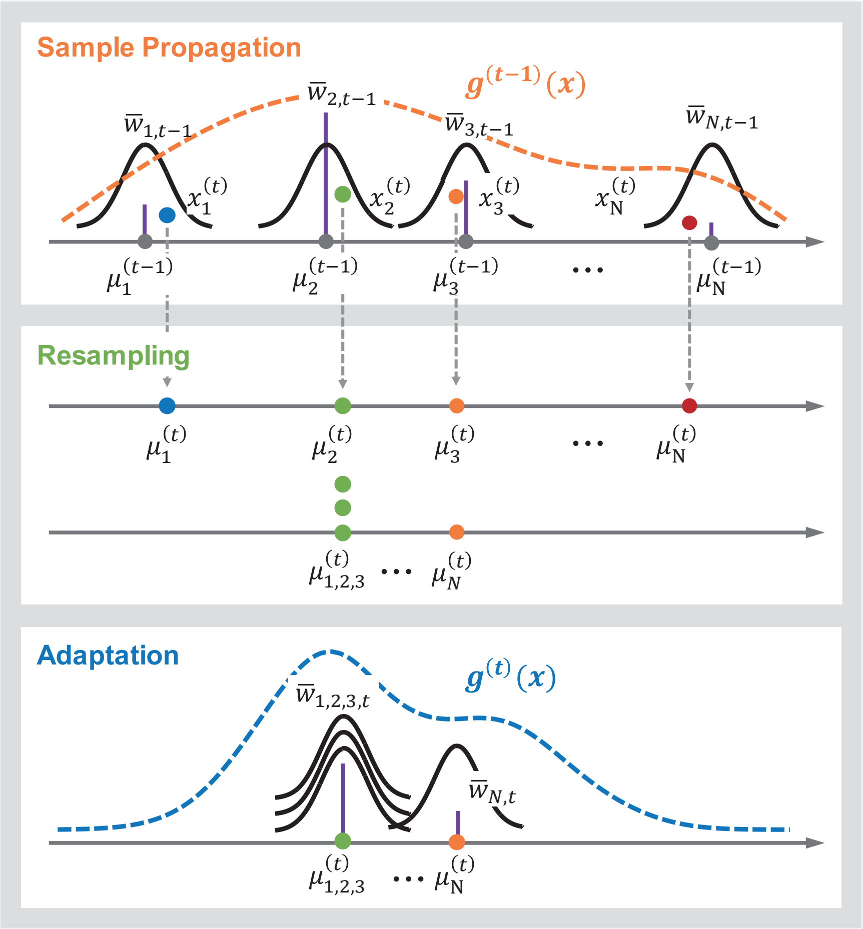


Figure 2.3: Illustrative figure of Resampling-based AIS at iteration t.

partial distribution with a new one by an accept/reject criterion. Figure 2.4 illustrates the improvement of $\{\bar{w}_{i,t}\}_{i=1}^{N}$ and sampling distribution $g^{(t)}(x)$ after two replacement operations. In practice, for each iteration, we repeat this procedure for $\mathcal{T}$ times, where $\mathcal{T}$ is a pre-defined parameter that can be tuned.

Algorithm 2 summarizes the detailed steps of our population MH sampler. At certain iteration t, we first select a "bad" partial distribution to be removed. The probability of selecting $\mu_k^{(t)}$ is proportional to the reciprocal of importance weight $\frac{1}{w_{i,t}}$. In the next generation step, we draw a new candidate location parameter μ_{new} from previous sampling distribution $g^{(t-1)}(x)$. The MH acceptance ratio is given by $\alpha = min\{1, \frac{\pi(\mu_{new})/g^{(t-1)}(\mu_{new})}{\pi(\mu_k^{(t)})/g^{(t-1)}(\mu_k^{(t)})}\}$. In the Statistics community [Bro98], the transition kernel $\mathcal{K}(\mu_{new}|\mu_k^{(t)})$ is defined as the conditional probability of replacing selected $\mu_k^{(t)}$ with μ_{new}, which is formulated as $\mathcal{K}(\mu_{new}|\mu_k^{(t)}) = g^{(t-1)}(\mu_{new})\alpha$. We

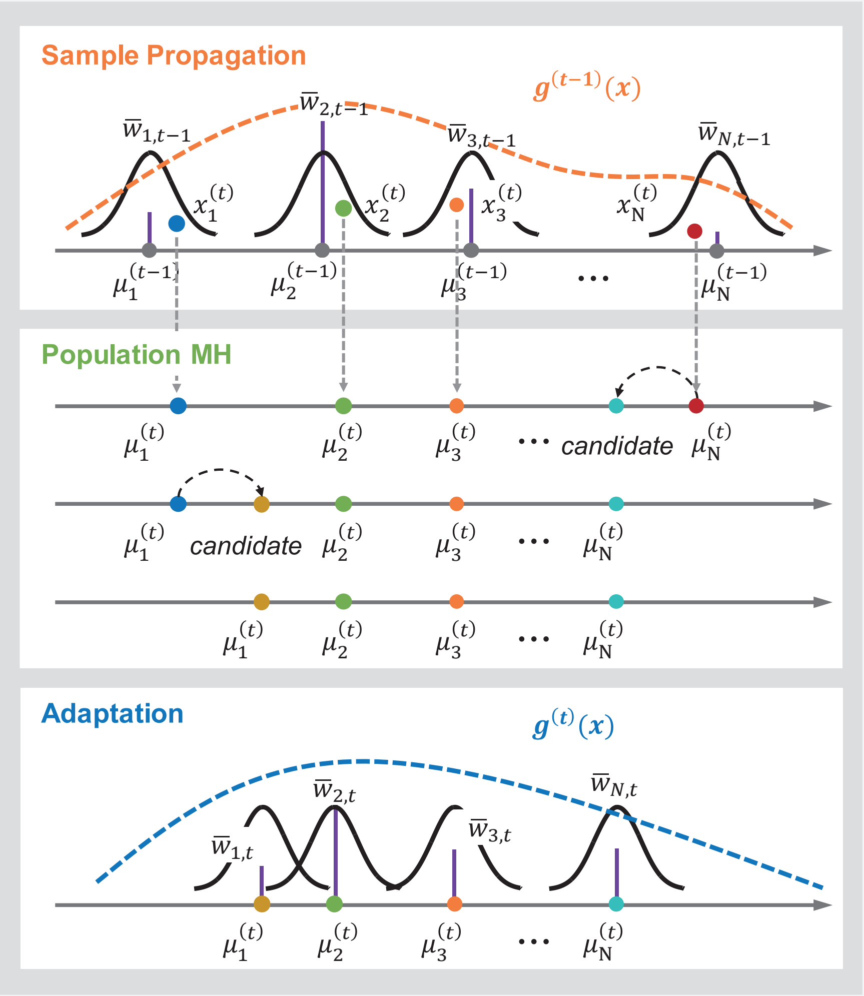


Figure 2.4: Illustrative figure of population MH-based AIS at iteration t.

use this transition kernel to satisfy the convergence criterion, which will be shown in Section V.

We notice that partial distributions with smaller weights are prone to be replaced, and thus sample points tend to move toward failure regions through a random walk. It is also worth noting that our population MH sampler will reduce to a standard MH when the number of population is one.

Algorithm 2: Population Metropolis-Hastings Sampler

```
Input: Current samples $\{x_i^{(t)}\}_{i=1}^N$, Normalized weights $\{w_{i,t}\}_{i=1}^N$.
Output: New location parameters set $\{\mu_i^{(t)}\}_{i=1}^N$.
Initialization:
Set $\{\mu_i^{(t)}\}_{i=1}^N = \{x_i^{(t)}\}_{i=1}^N$ and iteration index $\tau = 1$.
while $\tau < \mathcal{T}$ do
    Update iteration index $\tau = \tau + 1$ .
    1. Selection:
    Select a "bad" location parameter $\mu_k^{(t)}$ from a discrete PDF:
        $$\mu_k^{(t)} \sim \frac{1}{\sum_{i=1}^N \frac{1}{w_{i,t}}} \sum_{i=1}^N \frac{1}{w_{i,t}} \delta(\mu - \mu_i^{(t)})$$
     where $\delta(\mu)$ is impulse function.
    2. Generation:
    Draw a new candidate location parameter
        $$\mu_{new} \sim g^{(t-1)}(x)$$
    3. Replacing:
    Define acceptance ratio
        $$\alpha = min\{1, \frac{\pi(\mu_{new})/g^{(t-1)}(\mu_{new})}{\pi(\mu_k^{(t)})/g^{(t-1)}(\mu_k^{(t)})}\}$$
    Generate a random uniform number $r$ in $[0, 1]$.
    if $r \leq \alpha$ then
        accept new candidate $\mu_k^{(t)} = \mu_{new}$
    else
        reject new candidate $\mu_k^{(t)} = \mu_k^{(t)}$
    end
end
```

2.4.3 Comparison between Resampling and population MH

In this section, we compare Resampling method and population MH in terms of simulation cost and failure region coverage.

First, we notice that Resampling method exhibits inherit simplicity by "replicating" the partial distributions with higher importance weights and generating more sample points from these distributions. This procedure is easy to implement and does not require extra simulations. On the contrary, our population MH sampler seeks to "replace" the low-weight partial distributions with new ones according to a MH accept/reject rule. The calculation of acceptance ratio α involves the probability density of target distribution, which need SPICE simulation to evaluate. Specifically, for T AIS iterations, the total simulation runs of Resampling method are NT, while population MH requires $(N + \mathcal{T})T$ runs.

However, the major drawback of Resampling method is loss of sample diversity [EMLB17]. That is, the whole sample set is generated from one or a few partial distributions. Our proposed multinomial Resampling method disregards the spatial information of failure regions, and eliminates partial distributions in an uncensored way, which greatly reduces the diversity of sample points. We also notice that each Resampling step is irreversible: if a less important partial distribution is discarded in an intermediate iteration, it is most unlikely to revisit this region in the following iterations. For circuit cases with multiple failure regions, our Resampling method tends to aggregate the samples in a small region and may neglect other failure regions. On the contrary, for our population MH sampler, the weighting procedure is not deterministic but instead a stochastic process carried out by using a series of acceptance tests. In this way, the location parameter of new candidate distribution satisfies the failure event distribution, which guarantees full coverage of multiple failure regions.

2.5 AIS Estimator Analysis

2.5.1 Unbiasedness and Variance of AIS Estimator

We first derive that our AIS estimator $\hat{P}_{fail}$ keeps the unbiasedness of standard importance sampling estimator. It guarantees the expectation of estimation is equal to the failure probability P_{fail}.

$$\begin{aligned}
E[\hat{P}_{fail}] &= E[\frac{1}{TN}\sum_{t=1}^{T}\sum_{i=1}^{N} w_{i,t}] && (2.8)\\
&= \frac{1}{N}\sum_{i=1}^{N} E[\frac{f(x)I(x)}{\frac{1}{N}\sum_{j=1}^{N} q_j^{(t)}(x|\mu_j^{(t)},\Sigma_j)}] && (2.9)\\
&= \frac{1}{N}\sum_{i=1}^{N}\int \frac{f(x)I(x)}{\frac{1}{N}\sum_{j=1}^{N} q_j^{(t)}(x)} q_i^{(t)}(x)dx && (2.10)\\
&= \int \frac{f(x)I(x)}{\frac{1}{N}\sum_{j=1}^{N} q_j^{(t)}(x)}[\frac{1}{N}\sum_{i=1}^{N} q_i^{(t)}(x)]dx && (2.11)\\
&= \int f(x)I(x)dx = P_{fail} && (2.12)
\end{aligned}$$

Next, we demonstrate that proposed AIS estimator $\hat{P}_{fail}$ always has a lower or equal variance than the standard IS estimator $\hat{P}_{IS,fail}$. Mathematically, the variance of two estimators can be expressed as:

$$Var(\hat{P}_{IS,fail}) = \frac{1}{N^2}\sum_{i=1}^{N}(\int \frac{f^2(x)I^2(x)}{q_i(x)}dx - P_{fail}^2) \quad (2.13)$$

$$Var(\hat{P}_{fail}) = \frac{1}{N^2}\sum_{i=1}^{N}(\int \frac{f^2(x)I^2(x)}{\frac{1}{N}\sum_{j=1}^{N} q_j(x)}dx - P_{fail}^2) \quad (2.14)$$

By subtracting Equation (2.13) and (2.14), we prove that $Var(\hat{P}_{fail})$ is always smaller or equal by deriving the following inequality:

$$\sum_{i=1}^{N} \int \frac{f^2(x)I^2(x)}{\frac{1}{N}\sum_{j=1}^{N} q_j(x)} dx \tag{2.15}$$

$$= \sum_{i=1}^{N} \int \frac{f^2(x)I^2(x)}{\frac{N-1}{N}\frac{1}{N-1}\sum_{j=1}^{N-1} q_j(x) + \frac{1}{N}q_N(x)} dx \tag{2.16}$$

$$\leq \sum_{i=1}^{N} \int (\frac{(N-1)/N}{\frac{1}{N-1}\sum_{j=1}^{N-1} q_j(x)} + \frac{1/N}{q_N(x)}) f^2(x)I^2(x)dx \tag{2.17}$$

$$\leq \sum_{i=1}^{N} \int (\frac{1}{N}\sum_{j=1}^{N-1} \frac{1}{q_j(x)} + \frac{1}{N}\frac{1}{q_N(x)}) f^2(x)I^2(x)dx \tag{2.18}$$

$$= \sum_{i=1}^{N} \int (\frac{1}{N}\sum_{j=1}^{N} \frac{1}{q_j(x)}) f^2(x)I^2(x)dx \tag{2.19}$$

$$= \sum_{i=1}^{N} \int \frac{f^2(x)I^2(x)}{q_i(x)} dx \tag{2.20}$$

With the unbiasedness and less variance of estimation, we theoretically prove that AIS always has better performance than standard IS method. It is validated by the accuracy and efficiency comparison in Section VI.

2.5.2 Weight balancing

In this section, we investigate how our AIS estimator outperforms other IS based estimators in terms of weight balancing. As we explained in Section II, one major challenge of conventional IS technique occurs when the probability density discrepancy between original distribution $f(X)$ and sampling distribution $g(X)$ is too large. These overwhelming fluctuations in importance weight $w(X)$ can be prevented by our sampling distribution adaptation steps. To quantitatively analyze this feature, we introduce Effective Sample Size (ESS), which reflects the degree of weight degeneracy. ESS is extensively defined as [KLW94]:

$$ESS = \frac{1}{\sum_{i=1}^{N} (\bar{w}_{i,t})^2} \tag{2.21}$$

which involves only the normalized weights $\{\bar{w}_{i,t}\}_{i=1}^{N}$.

As we have introduced in Section IV, our Resampling scheme generates more samples from partial distributions with larger weights and eliminates distributions with smaller weights. Thus $\{\bar{w}_{i,t}\}_{i=1}^{N}$ is naturally balanced. On the other hand, our population MH sampler constructs a stochastic process, which is equivalent to draw samples from the failure event distribution. Ideally, optimal $\{\bar{w}_{i,t}\}_{i=1}^{N}$ satisfy $w^* = \frac{1}{N}$. An alternative view is that our entire AIS sample set $\{x_i^{(t)}, w_{i,t}\}_{i=1}^{NT}$ has equivalent influence on the estimator $\hat{P}_{fail} = \frac{1}{TN}\sum_{t=1}^{T}\sum_{i=1}^{N} w_{i,t}$. To this end, both of sampling distribution adaption methods can be applied to prevent weight degeneracy and maintain the robustness of AIS estimator.

2.5.3 Convergence Criteria

The proposed convergence criteria, varied with different distribution adaptation schemes, determine the stability of failure probability estimation.

For Resampling-based AIS, the weighting procedure is provided by a deterministic function. Therefore, the consistency of convergence is naturally inherent [LBD15].

For MH-based AIS, the location of partial distribution $\{\mu_i^{(t)}\}_{i=1}^{N}$ is generated from a stochastic process, by applying a series of transition kernel $\mathcal{K}(\mu_{new}|\mu_k^{(t)})$ to replace $\mu_k^{(t)}$ conditionally with μ_{new}. The goal is to prove that the probability density of drawing new candidate μ_{new} is equivalent to sampling from failure event distribution $\pi(x)$, written as $p(\mu_{new}) = \pi(\mu_{new})$.

In order to derive the convergence criterion, we first introduce the Lemma of detailed balance condition:

$$\mathcal{K}(\mu_{new}|\mu_k^{(t)})\pi(\mu_k^{(t)}) = \mathcal{K}(\mu_k^{(t)}|\mu_{new})\pi(\mu_{new}) \tag{2.22}$$

This Lemma indicates that the transition rate from $\mu_k^{(t)}$ to μ_{new} is equal to the transition rate from μ_{new} to $\mu_k^{(t)}$ in a stationary state. The proof of detailed balance condition is given by substituting the expression of $\mathcal{K}(\mu_{new}|\mu_k^{(t)})$ in Equation (2.22), we have

$$\mathcal{K}(\mu_{new}|\mu_k^{(t)})\pi(\mu_k^{(t)}) \tag{2.23}$$

$$= cg^{(t-1)}(\mu_{new})min\{1, \frac{\pi(\mu_{new})/g^{(t-1)}(\mu_{new})}{\pi(\mu_k^{(t)})/g^{(t-1)}(\mu_k^{(t)})}\}\pi(\mu_k^{(t)}) \tag{2.24}$$

$$= cg^{(t-1)}(\mu_k^{(t)})min\{1, \frac{\pi(\mu_k^{(t)})/g^{(t-1)}(\mu_k^{(t)})}{\pi(\mu_{new})/g^{(t-1)}(\mu_{new})}\}\pi(\mu_{new}) \tag{2.25}$$

$$= \mathcal{K}(\mu_k^{(t)}|\mu_{new})\pi(\mu_{new}) \tag{2.26}$$

Next, we derive the convergence criterion

$$p(\mu_{new}) = \int \mathcal{K}(\mu_{new}|\mu_k^{(t)})p(\mu_k^{(t)})d\mu_k^{(t)} \tag{2.27}$$

$$= \int \mathcal{K}(\mu_{new}|\mu_k^{(t)})\pi(\mu_k^{(t)})d\mu_k^{(t)} \tag{2.28}$$

$$= \int \mathcal{K}(\mu_k^{(t)}|\mu_{new})\pi(\mu_{new})d\mu_k^{(t)} \tag{2.29}$$

$$= \pi(\mu_{new}) \int \mathcal{K}(\mu_k^{(t)}|\mu_{new})d\mu_k^{(t)} \tag{2.30}$$

$$= \pi(\mu_{new}) \tag{2.31}$$

We notice that this convergence criterion requires the assumption of "ergodic" property of population MH, which means that sample points can visit any state in the failure regions. It ensures that our MH sampler will converge to a stationary state. That is, the PDF of $p(\mu_{new})$ will approach $\pi(\mu_{new})$ as t$\rightarrow \infty$.

In conclusion, both of our sampling distribution adjustment methods converge to the stable target distribution $\pi(x)$ when sufficient simulations are provided. These methods motivate the adaptation mechanism of our AIS algorithm. Compared with conventional static IS estimator, the setup time for presampling can be drastically reduced. In practice, we truncate this sampling procedure with limited simulations, while preserving failure probability estimation within a pre-decided confidence interval.

2.6 Experiment Result

The proposed Resampling-based AIS (R-AIS) and population MH-based AIS (MH-AIS) methods are validated on both memory and analog circuits. They are first evaluated on a typical 6T SRAM bit cell circuit with 54 variables, then on a two-stage amplifier circuit with 126 variables for analog yield analysis. We implement MC as ground truth for accuracy comparison. We also perform different methods, including High Dimensional Importance Sampling (HDIS) [WGCH14], Hyperspherical Clustering and Sampling (HSCS) [WBH16] for comparison purpose. We run HSPICE with SMIC 40nm model. All the experiments are performed on a Linux server with Intel Xeon X5675 CPU @3.07 GHz and 94 GB RAM.

2.6.1 Experiments on 6T SRAM Bit Cell

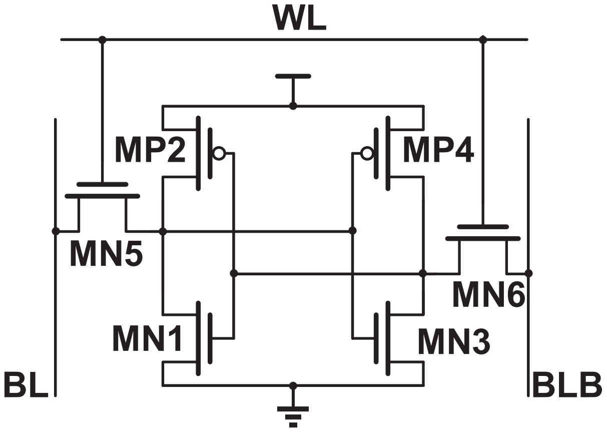


Figure 2.5: The schematic of typical 6T SRAM bit cell.

In this experiment, a typical 6-transistor SRAM bit cell is used to test the accuracy and efficiency of the proposed AIS methods. Figure 2.5 shows the schematic of the SRAM bit cell. Four transistors $MN1$, $MP2$, $MN3$ and $MP4$ form two cross-coupled inverters and utilize two steady states to store either “0” or “1”; transistor $MN5$ and $MN6$ control accessing to the storage cell during read, write or standby operations. Without loss of generality, we take SRAM read failure as the failure mechanism of interest. During read operation, after the bit line BL and BLB is precharged, the bit line voltage V_{BL} and V_{BLB} decrease due

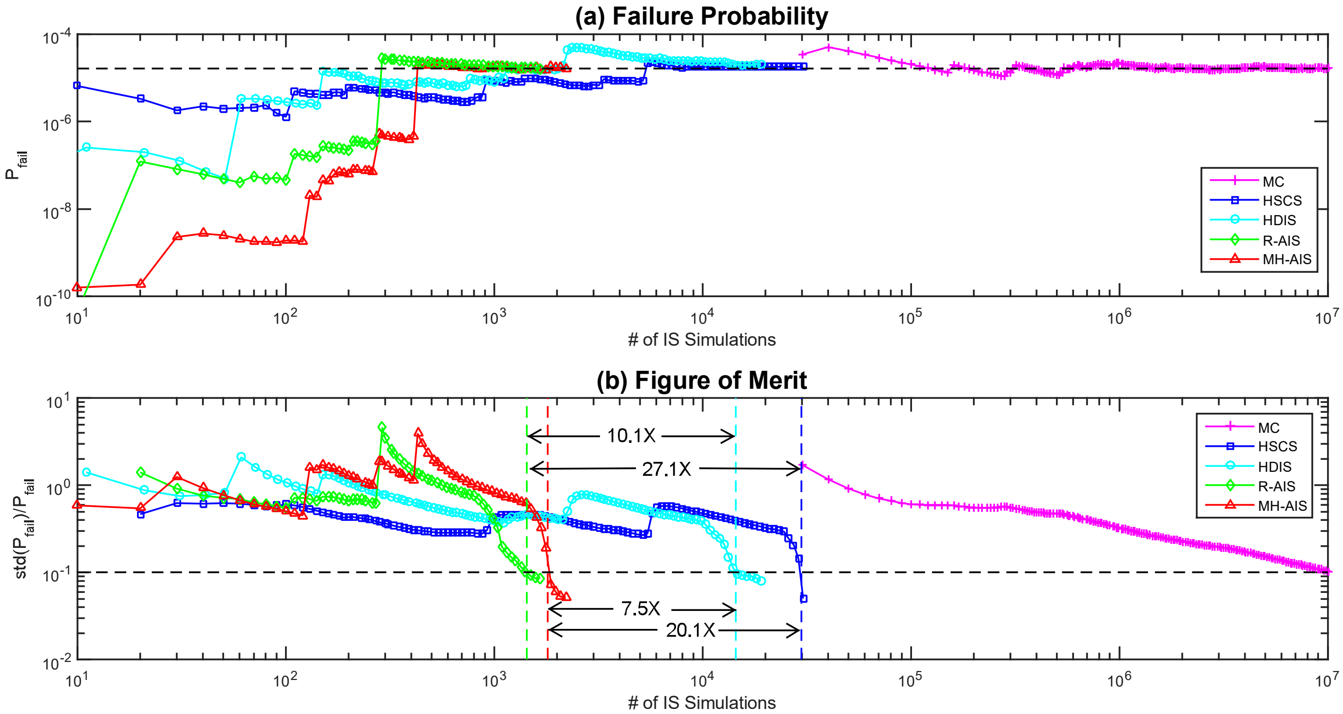


Figure 2.6: Evolution comparison of failure prob. and FOM vs. the number of IS simulations. Efficiency comparison in Figure 2.6(b) considers only IS simulations (without Presampling simulations).

to the current I_{BL} and I_{BLB} through the access transistors and pull-down transistors. The SRAM read failure occurs when the voltage difference between BL and BLB is too small to be captured by sense amplifier at the end of read operation. The process variations of each transistor are modeled by 9 parameters including threshold voltage V_{th0}, gate oxide thickness t_{ox}, Mobility μ_0, etc. In total, the number of parameters is 54. And all these parameters are treated as mutually independent and standard normal [DM03].

2.6.1.1 Accuracy Comparison

In order to evaluate the accuracy, Figure of Merit, ρ, is employed to characterize the accuracy and confidence of our estimation. It is defined as:

$$\rho = \frac{\sqrt{\sigma^2_{\hat{P}_{fail}}}}{\hat{P}_{fail}} \tag{2.32}$$

Table 2.1: Accuracy and efficiency comparison on 54-dimensional SRAM bit cell circuit

	MC	HSCS	HDIS	R-AIS	MH-AIS
Failure prob.(error)	1.66e-5(0%)	1.77e-5(6.6%)	2.01e-5(22.1%)	1.67e-5(0.6%)	1.69e-5(1.8%)
Presampling	0	4500	8000	3000	3000
IS	1e7	38750	14480	1430	1925
Total (speedup)	1e7(1×)	43250(231×)	22480(445×)	4430(2257×)	4925(2030×)

where $\hat{P}_{fail}$ denotes the estimation of failure probability and $\sigma_{\hat{P}_{fail}}$ denotes the standard deviation of $\hat{P}_{fail}$. In fact, we can declare one estimate is $(1-\epsilon)100\%$ accurate with $(1-\delta)100\%$ confidence when $\rho < \epsilon\sqrt{log(1/\delta)}$. Here we use $\rho < 0.1$ to indicate the estimation reaches a steady state with 90% confidence interval. ρ has been extensively used in the literature [WGCH14], [WBH16].

The evolutions of failure probability estimation and calculation of FOM are plotted in Figure 2.6. From Figure 2.6(a), we notice that the estimation of HDIS, HSCS, R-AIS and MH-AIS methods all converge. The stable estimation value is reached when the FOM is smaller than 0.1, which is denoted by the dashed line in Figure 2.6(b). The numerical comparison is illustrated in Table 2.1. The ground truth MC result is 1.66e-5 (4.26σ). Among these methods, our proposed R-AIS and MH-AIS are more accurate with less than 2% relative error, while the results of HSCS and HDIS have 6.6% and 22.1% relative error, respectively. This stable estimation is guaranteed by our carefully designed convergence criterion.

2.6.1.2 Efficiency Comparison

Figure 2.6(b) illustrates the efficiency of our proposed algorithm. We notice that both R-AIS and MH-AIS have faster convergence speed compared with other algorithms. It is attributed to their dynamic search process, which is far more efficient than sampling from static distribution. Specifically, in the Impotance Sampling (IS) stage, R-AIS and MH-AIS method can reach a steady state with 90% confidence interval within only 1430 and 1925 IS simulations. We notice that MH-AIS is slightly slower because it requires extra simulations to evaluate the weight of new candidates. On the contrary, HSCS and HDIS need 38750 and 14480 IS simulations to reach the same FOM value, respectively. Thus our methods can obtain 7-27× speedup w.r.t HSCS and HDIS method in the IS stage.

Apart from IS simulations, we note that all importance sampling based approaches need extra presampling simulations to initialize the sampling distribution. To be specific, as indicated in Table 2.1, two AIS methods need only 3000 presampling simulations, while HSCS and HDIS need 4500 and 8000 times.

In total, Table 2.1 reveals that R-AIS achieves 2257× speedup over Monte Carlo, 9.8× over HSCS, and 5.1× over HDIS, while MH-AIS is 2030× faster than Monte Carlo, 8.8× than HSCS, and 4.6× than HDIS.

2.6.2 Experiments on Two stage Amplifier

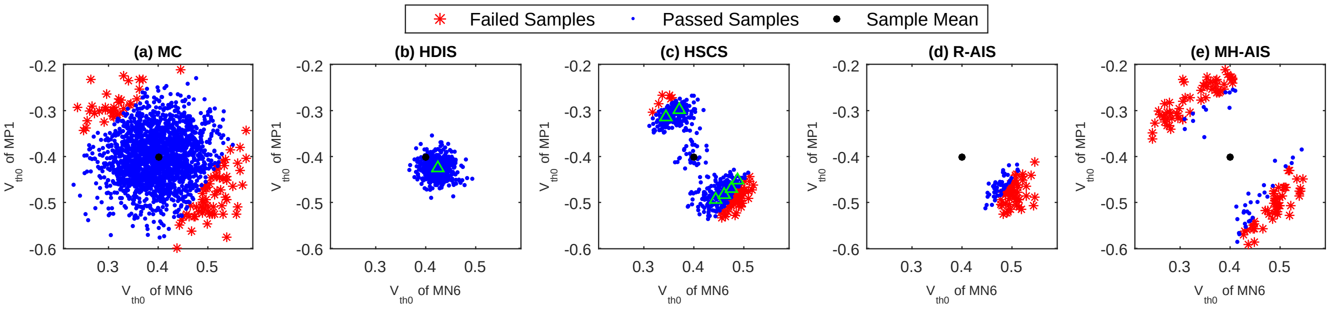


Figure 2.7: Multiple failure region coverage test (failed samples/passed samples/sample mean are colored separately)

Next, we evaluate the performance of proposed AIS methods on analog circuit. A simplified schematic of the two-stage operational transimpedance amplifier (OTA) consisting of master-slave structure for low supply voltage application is presented in Figure 2.8. The slave stage consists of the tail current transistor ($MP5$), the differential pair ($MP1$ and $MP2$) and the current-mirror load ($MN1$ and $MN2$). The master stage should copy the tail current source and the transconductance transistor of the slave stage (i.e. $MP3$, $MP4$, $MP6$ and $MN3$ are the replicates of $MP1$, $MP2$, $MP5$ and $MN1$). In order to save voltage

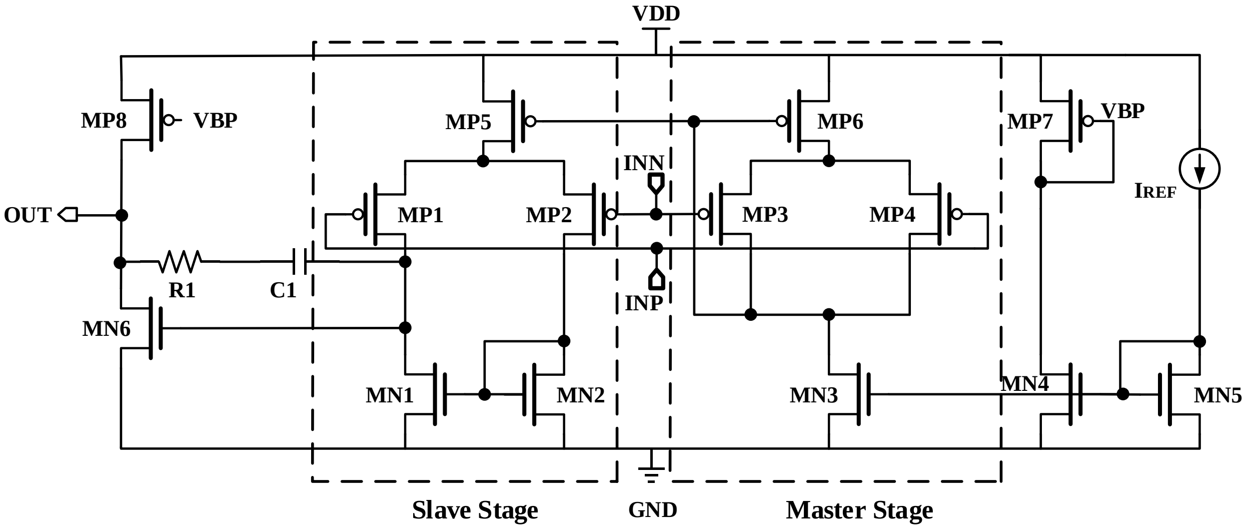


Figure 2.8: The schematic of two-stage amplifier circuit.

margin, transistor $MP5$ and $MP6$ should operate in linear region.

Compared with the experiment on 6T SRAM, the estimation of P_{fail} for OTA circuit is more troublesome because it has various failure mechanisms which lead to multiple failure regions. In our experimental setting, we consider four performance specifications, including voltage gain margin, gain bandwidth, phase margin and 3dB bandwidth. For this amplifier circuit, there are in total 126 variation parameters.

Table 2.2: Accuracy and efficiency comparison on 126-dimensional two-stage amplifier circuit

	MC	HSCS	R-AIS	MH-AIS
Failure prob.(error)	1.5e-3(0%)	1.26e-3(16%)	7.82e-4(47.9%)	1.52e-3(1.3%)
Presampling # sim.	0	7000	3000	3000
Importance Sampling # sim.	458700	49290	956	2093
Total # sim. (speedup)	458700(1×)	56290(8×)	3956(116×)	5093(90×)

2.6.2.1 Comparison of Visualized Failure Regions

In order to investigate the different performance among aforementioned methods, we project the sample points onto two most important dimensions, V_{th0} of $MP1$ and V_{th0} of $MN6$, generating a visualized scatter plot. Figure 2.7 demonstrates the multiple failure regions in 2D parametric space for different methods. Two disjoint failure regions are clearly displayed in Figure 2.7(a) with MC sampling. We notice that it consists of a major failure region in the bottom-right corner and a minor one in the upper-left.

In Figure 2.7(b), we notice that HDIS cannot capture any failure samples with limited IS simulations. The reason is that mean-shift IS cannot deal with multi-failure-region circuits. HDIS locates the shifted sample mean (green triangle) at the centroid of failure samples drawn in the presampling stage. For circuits with multiple failure regions, it is infeasible to find a single sample mean that is close to all failure regions.

Figure 2.7(c) shows that HSCS method with multiple mean-shift vectors can detect both of the failure regions. It is mainly attributed to its directional K-means algorithm to group the failure samples and take multiple min-norm points as target of IS shift vectors. As a static mixture IS method, HSCS is highly dependent on presampling to determine the sampling distribution. It is, however, challenging to detect the failure region boundaries when the number of simulations is limited. With biased shifted means, we notice that HSCS requires more simulations and has larger relative error.

R-AIS is able to search for the failure regions through a resampling procedure, and the majority of samples are located around important failure region boundaries. However, R-AIS does not exhibit very stable performance. Figure 2.7(d) depicts a special case that R-AIS fails to find both of the failure regions. Due to the reduction of sample diversity, we notice that the minor region in the upper-left disappears during the resampling iterations.

In Figure 2.7(e), MH-AIS achieves best performance among these methods. It can successfully search for both failure regions with only a few iterations. The adaptation of population MH sampler provides the flexibility to search for failure regions in the entire parametric space, which makes it suitable for more complex circuits with multiple failure regions.

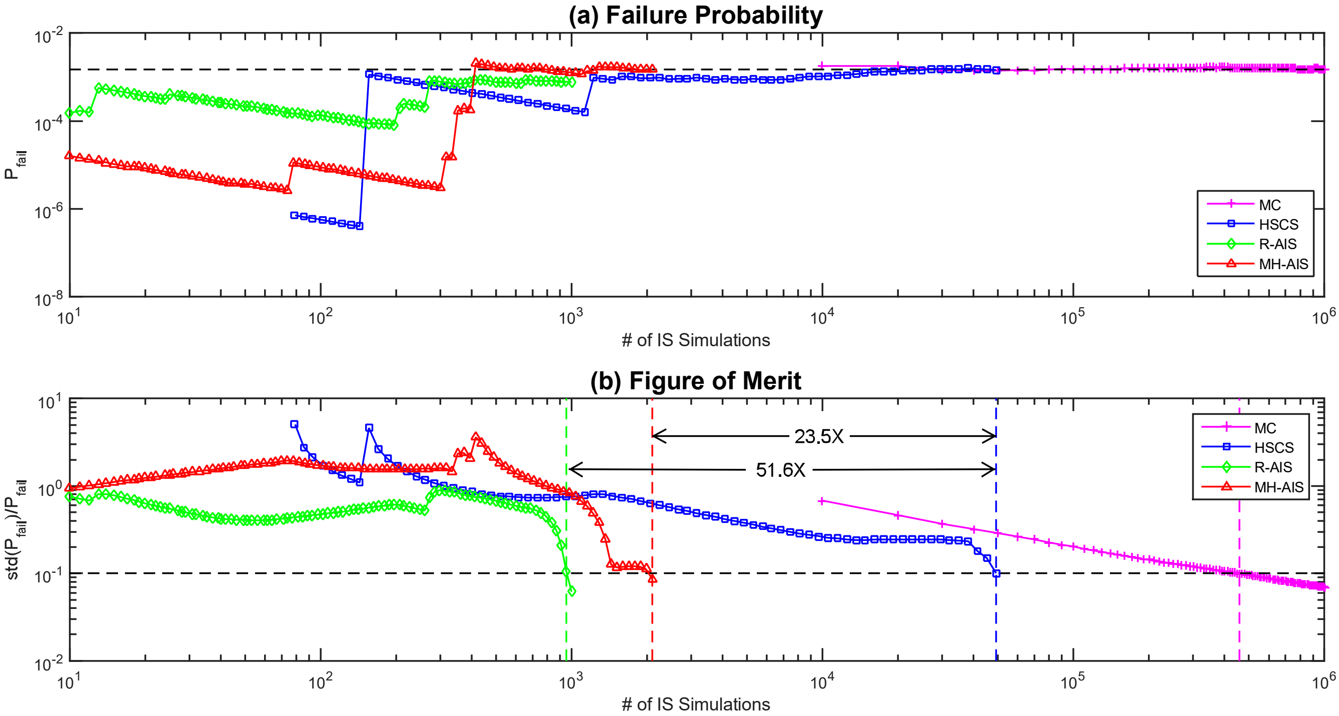


Figure 2.9: Evolution comparison of failure prob. and FOM vs. the number of IS simulations. Efficiency comparison in Figure 2.9(b) considers only IS simulations (without Presampling simulations).

2.6.2.2 Accuracy and Efficiency Comparison

In order to validate the accuracy of our algorithm, we plot the evolution of failure probability estimation and FOM vs. number of IS simulations in Figure 2.9. To generate ground truth, MC takes 4.59e5 simulations to obtain confident estimation of P_{fail} at 1.5e-3 (3.18σ). We notice that only HSCS and MH-AIS methods can successfully estimate failure probability, while HDIS fails to discover any failure sample and R-AIS converges to a wrong failure probability. The FOM of our MH-AIS method descends to 0.1 within 5093 simulations (including 3000 presampling simulations), which represents a 11× speedup over HSCS and 90× speedup over MC method.

2.7 Conclusion

In this chapter, we present an adaptive importance sampling method to efficiently estimate the rare event failure probability of memory and analog circuits. AIS first applies hyperspherical presampling to construct initial sampling distribution. We also develop two distinct sampling distribution adaption schemes to search for failure regions and balance sample weights. Experiments on SRAM bit cell circuit indicate that AIS method can achieve three order of magnitude speedup over MC and 4-10× over other state-of-the-art methods. On another two-stage amplifier circuit, AIS is 90× faster than MC method, while other approaches fail to converge to correct failure probability.

CHAPTER 3

Adaptive Clustering and Sampling for Fast Yield Analysis

3.1 Summary

In this chapter, we present an accurate and efficient algorithm based on Adaptive Clustering and Sampling (ACS) method to estimate the failure rate of high-dimensional and multi-failure-region circuit cases. The basic idea of the algorithm is to cluster failure samples and build global sampling distribution at each iteration. Specifically, in clustering step, we propose a multi-cone clustering method, which partitions the parametric space and clusters failure samples. Then global sampling distribution is constructed from a set of weighted Gaussian distributions. Next, we calculate importance weight for each sample based on the discrepancy between sampling distribution and target distribution. Failure probability is updated at the end of each iteration. This clustering and sampling procedure proceeds iteratively until all the failure regions are covered.

Our main contribution is summarized in three aspects: first, we initialize our ACS algorithm with hyperspherical presampling method, which reduces dimension by sampling from a set of hyperspherical surfaces. Second, we propose an adaptive scheme to explore high-dimensional space and search for failure regions. As iteration continues, each partial IS estimator provides a better estimation, and global sampling distribution will tilt toward failure region. Moreover, our estimator is adapted parallelly in different directions, which can effectively improve sample diversity.

3.2 Algorithm Description

Algorithm 3 summarizes the main steps of proposed ACS method. The objective of this algorithm is to iteratively explore multiple failure regions in different directions and update failure rate at the end of each iteration. In the initialization step, we collect M samples $\{X_i^{(0)}\}_{i=1}^M$ by hyperspherical presampling, and group these samples into k clusters $\{C_j^{(0)}\}_{j=1}^k$ using our multi-cone clustering algorithm. At each iteration t, we construct a local sampling distribution $g_j^{(t-1)}(x)$ in each cluster $C_j^{(t-1)}$ based on the samples that assigned to it. Our global sampling distribution $g^{(t-1)}(x)$ is built from a weighted mixture of all the local sampling distributions $\{g_j^{(t-1)}(x)\}_{j=1}^k$. Next we generate M new samples $\{X_i^{(t)}\}_{i=1}^M$ from $g^{(t-1)}(x)$, and calibrate the cluster sets $\{C_j^{(t)}\}_{j=1}^k$. At the end of each iteration, we update failure probability estimation by averaging all the partial IS estimators $\hat{P}_{fail,t} = \frac{1}{tM} \sum_{l=1}^t \sum_{i=1}^M w_{i,l}$ up to present. This iteration proceeds iteratively until our estimation converge to certain confidence interval.

3.2.1 Hyperspherical Presampling

One major challenge in implementing our ACS algorithm is to generate some initial samples that can locate multiple failure regions. In practice, an effective initialization method can help improve sample diversity, convergence speed, and capability to explore parametric space. It is quite difficult to recover from a set of poor starting samples, and the adaptation may converge to a local optimum in the parametric space.

```
Algorithm 3: ACS Algorithm
Initialization:
1. Set iteration index $t = 0$, generate initial failure sample set $\{X_i^{(0)}\}_{i=1}^{M}$.
2. Assign failure samples $\{X_i^{(0)}\}_{i=1}^{M}$ to cluster set $\{C_j^{(0)}\}_{j=1}^{k}$.
Repeat
Update iteration index $t = t + 1$.
    1. Cluster sampling distribution:
    (a) Construct Gaussian distribution as sample proposal:
            $q_i^{(t-1)}(x) = N(X_i^{(t-1)}, \Sigma)$
    (b) Calculate probability density for $N_j^{(t)}$ failure samples:
            $\beta_{i,t-1} = f(X_i^{(t-1)})I(X_i^{(t-1)}) \qquad i = 1, ..., N_j^{(t)}.$
    (c) Construct local distribution $g_j^{(t-1)}(x)$:
            $g_j^{(t-1)}(x) = \frac{1}{\sum_{i=1}^{N_j} \beta_{i,t-1}} \sum_{i=1}^{N_j} \beta_{i,t-1} \cdot q_i^{(t-1)}(x)$
    2. Sample propagation:
    Generate M samples from global distribution $g^{(t-1)}(x)$:
            $X_i^{(t)} \sim g^{(t-1)}(x) = \frac{1}{\sum_{i=1}^{M} \beta_{i,t-1}} \sum_{j=1}^{k} \sum_{X_i \in C_j^{(t)}} \beta_{i,t-1} \cdot g_j^{(t-1)}(x)$
    3. Cluster calibration:
    Re-cluster failure samples $\{X_i^{(t)}\}_{i=1}^{M}$ into new set$\{C_j^{(t)}\}_{j=1}^{k}$ and update $\{N_j^{(t)}\}_{j=1}^{k}$.
    4. Failure probability calculation:
    (a) Compute incremental importance weight:
            $w_{i,t} = \frac{\pi(x)}{g^{(t-1)}(x)} = \frac{f(x)I(x)}{g^{(t-1)}(x)} \qquad i = 1, ..., M.$
    (b) Update unbiased estimator using all samples up to present iteration:
            $\hat{P}_{fail,t} = \frac{1}{tM} \sum_{l=1}^{t} \sum_{i=1}^{M} w_{i,l}$
Until Relative standard deviation (FOM): $\rho = \frac{\sqrt{\sigma^2_{\hat{P}_{fail}}}}{\hat{P}_{fail}} \leq 0.1$
```

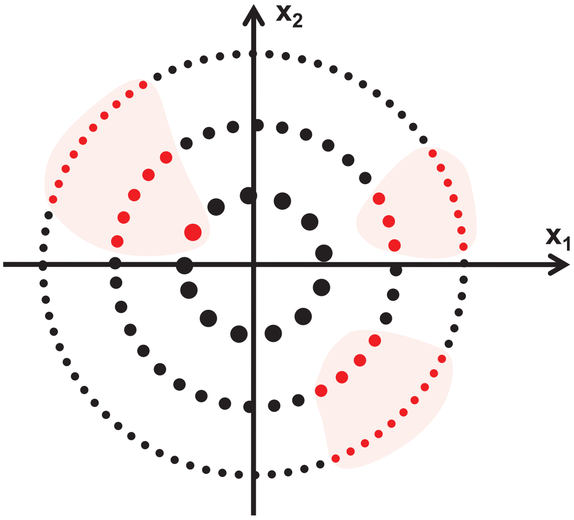


Figure 3.1: Incremental hyperspherical presampling (for finding initial samples to enable local exploration)

In order to develop a good initialization, as demonstrated in Figure 3.1, we implement a hyperspherical presampling procedure. This procedure starts with circuit nominal distribution, which indicates a unit hypersphere in parametric space. The radius of this unit hypersphere denotes the variance of circuit parameters. To cover multiple failure regions, we gradually increase the radius of hypersphere until M failure samples are captured. Our presampling method is a dimension reduction process by restricting the samples on a union of hyperspherical surfaces. This method can accelerate the exploration in the high dimensional space while maintaining sample diversity.

In Algorithm 1, the number of initial failure samples, M, can be arbitrarily user specified. We note that there exists a trade-off between estimation accuracy and algorithm complexity: the larger M is, the higher estimation accuracy we can achieve while sacrificing simulation runtime. In our experiments, we collect 100 samples on each hypersphere until 10% or more samples fail.

3.2.2 Multi-Cone Clustering

After we collect a set of failure samples located on several discrete hyperspherical surfaces with different radius, we need to cluster these samples with the boundary of multiple disjoint failure regions. Conventional clustering algorithms apply techniques such as graph-based methods, density-based methods and boundary-based methods. They group sample points into optimal clusters by evaluating the Euclidean distance between sample pairs, as defined in (8).

$$EuclideanDistance(X^{(1)}, X^{(2)}) = \|X^{(1)} - X^{(2)}\| \tag{3.1}$$

However, clustering samples that are randomly distributed in high dimensional open space is challenging. As the dimension of parametric space increases, the ratio of Euclidean distance between nearest and farthest neighbors is closer to 1. In such cases, the nearest neighbor problem becomes ill-defined and qualitative clustering methods are unapproachable.

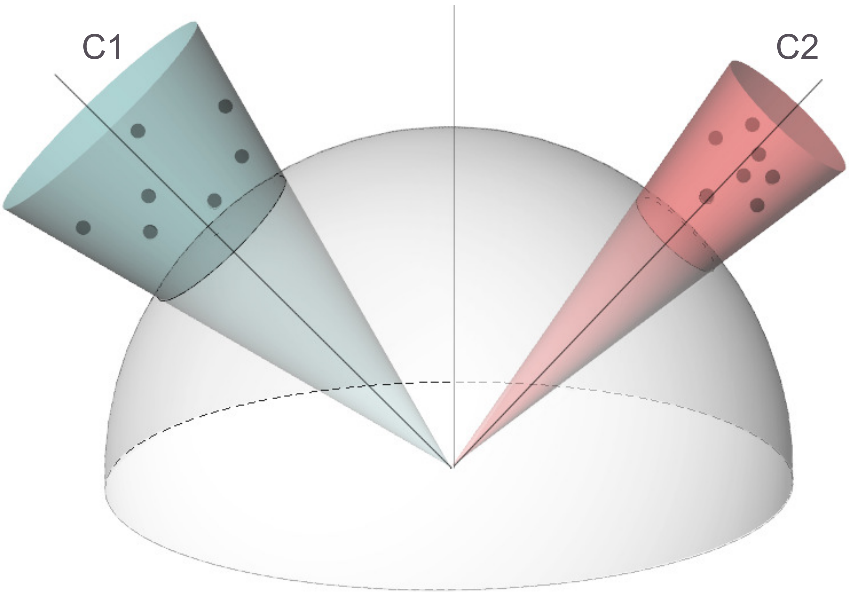


Figure 3.2: Partition the space into non-overlapping cones along radial directions.

Alternatively, at each iteration, we cluster the sample points based on their directions rather than Euclidean distance. We partition the high-dimensional space into a union of disjoint cones, while maintaining whole space coverage. For each cone-shaped subspace, we build a nonlinear mapping from sample space to feature space. As illustrated in Figure 3.2, all distinct sample points in each cone are projected to the unit hypersphere surface in the radial direction. We note that the location of images projected on the unit hypersphere describes the direction of sample points. The distance metric of each sample pair is evaluated by the cosine distance of direction vectors, as defined in (9).

$$CosineDistance(X^{(1)}, X^{(2)}) = 1 - \frac{X^{(1)} \cdot X^{(2)}}{\|X^{(1)}\|\|X^{(2)}\|} \tag{3.2}$$

As shown in Algorithm 4, a k-means based algorithm is implemented to cluster the failure samples $\{X_i\}_{i=1}^{M}$ according to their direction. The algorithm proceeds as follows. We first generate k unit length vectors $\{V_j\}_{j=1}^{k}$ as initial direction vectors arbitrarily, where k is a user specified parameter. Each sample X_i is then assigned to the closest cluster C_{λ_i} according to the cosine distance. Next, the direction vector of each cluster is updated to the average of associated sample vectors. This assignment and update procedure is repeated until cluster membership stabilizes.

Although k-means algorithm has advantages of its simplicity to implement and efficiency, it searches for clusters in a greedy fashion, which makes it sensitive to bad initialization and outliers. In our experiment, we start from multiple set of initial direction vectors, and minimize sum of cosine distance as error function. Thus, our clustering result is more robust and more prone to converge to global optimum rather than local optimum.

We also note that the number of clusters, k, can be tuned to improve the effectiveness of our k-means algorithm. Larger k always improves the cluster cohesiveness by decreasing squared error, but this comes at the expense of higher computational cost. We explore this trade-off by utilizing different k values. In practice, k is fixed as $\sqrt{M}$, where M is the number of failure samples to be clustered.

Algorithm 4: Multi-cone Clustering Algorithm

```
Input:
    Failure sample set: $\{X_i\}_{i=1}^{M}$
    Initial cluster number: $k$
Output:
    Cluster label for samples: $\{\lambda_i\}_{i=1}^{M}$
Initialization:
    Randomly initialize $k$ unit length direction vectors
    $\{V_j\}_{j=1}^{k}$ and corresponding empty clusters $\{C_j\}_{j=1}^{k}$.
Repeat
  1. For each sample $X_i$, calculate cosine distance with all the direction vectors $\{V_j\}_{j=1}^{k}$:
         $CosineDistance(X_i, V_j) = 1 - \frac{X_i \cdot V_j}{\|X_i\| \|V_j\|}$.
     Update $\lambda_i = \underset{j}{\operatorname{argmin}} \, CosineDistance(X_i, V_j)$ and assign $X_i$ to cluster $C_{\lambda_i}$.
  2. For each cluster $C_j$, update its direction vector:
         $V_j = \frac{1}{|C_j|} \sum_{X \in C_j} X$.
     where $|C_j|$ denotes the number of samples in cluster $C_j$.
Until
    Sample labels $\{\lambda_i\}_{i=1}^{M}$ remain unchanged.
```

3.3 ACS Estimator Analysis

3.3.1 Unbiasedness of ACS Estimator

In this section, we prove that our ACS estimator $\hat{P}_{fail}$ is unbiased between iterations. The unbiasedness of estimator is evaluated by its expected value. It guarantees the estimation built from random measure $\{X_i^{(t)}, w_{i,t}\}_{i=1}^{M}$ is consistent, and it converges to failure probability P_{fail}.

$$E[\hat{P}_{fail}] = E[\frac{1}{tM}\sum_{l=1}^{t}\sum_{i=1}^{M} w_{i,l}] \tag{3.3}$$

$$= \frac{1}{tM}\sum_{l=1}^{t}\sum_{i=1}^{M} E[\frac{f(x)I(x)}{g^{(l-1)}(x)}] \tag{3.4}$$

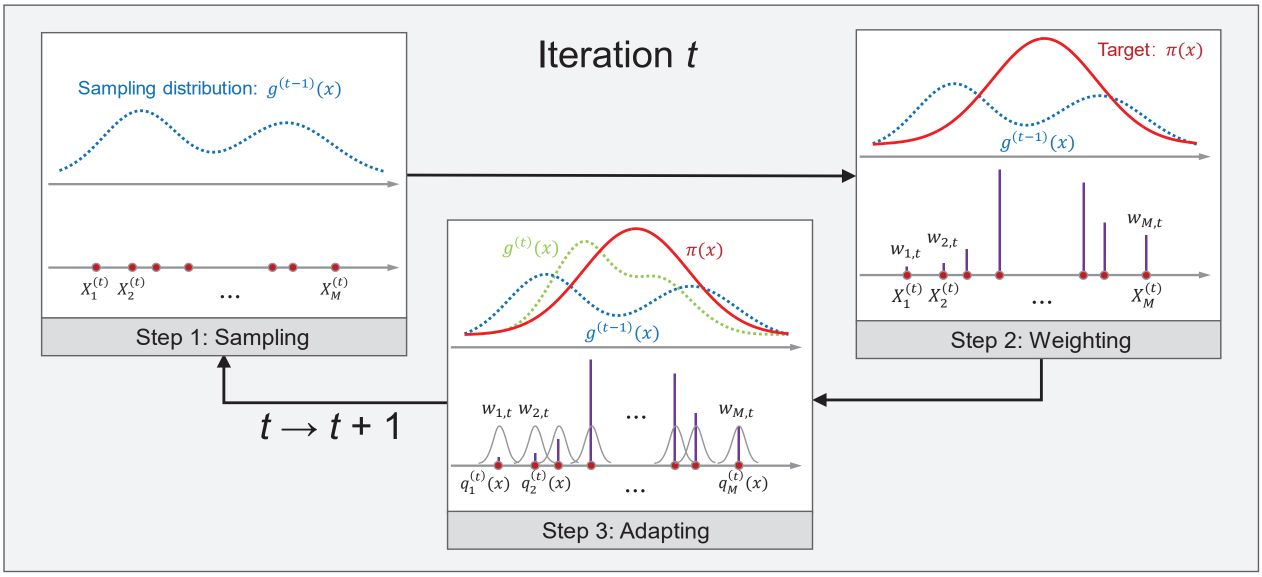
$$= \frac{1}{tM}\sum_{l=1}^{t}\sum_{i=1}^{M} \int \frac{f(x)I(x)}{g^{(l-1)}(x)} g^{(l-1)}(x)dx \tag{3.5}$$

$$= \frac{1}{tM}\sum_{l=1}^{t}\sum_{i=1}^{M} \int f(x)I(x)dx = P_{fail} \tag{3.6}$$

3.3.2 Adaptation of ACS Estimator

The ACS methodology is based on an iterative process which proceeds parallelly in multiple disjoint clusters. With the adaptation of ACS estimator, our sampling distribution $g^{(t)}(x)$ gradually evolves to accurately approximate the target probability density $\pi(x)$. As demonstrated in Algorithm 1, this procedure consists of three main stages: generating samples from sampling distribution (sampling), calculation of the incremental importance weight for each of the samples (weighting) and updating the parameters to define the new sampling distribution for next iteration (adapting). Figure 3.3 shows the flow diagram of the stages in ACS.

The adaptive mechanism of ACS is driven by the uncertainty in the partial IS estimators, which can be quantified by their variance. To be specific, each sample forms a sample proposal $q_i^{(t)}(x)$ that can describe local features of the target distribution $\pi(x)$. In order to obtain a discrete probability distribution that approximates target distribution, we introduce



Iteration t
Sampling distribution: $g^{(t-1)}(x)$
$X_1^{(t)}$ $X_2^{(t)}$ … $X_M^{(t)}$
Step 1: Sampling
Target: $\pi(x)$
$g^{(t-1)}(x)$
$w_{1,t}$ $w_{2,t}$ $w_{M,t}$
$X_1^{(t)}$ $X_2^{(t)}$ … $X_M^{(t)}$
Step 2: Weighting
$g^{(t)}(x)$
$\pi(x)$
$g^{(t-1)}(x)$
$w_{1,t}$ $w_{2,t}$ … $w_{M,t}$
$q_1^{(t)}(x)$ $q_2^{(t)}(x)$ … $q_M^{(t)}(x)$
Step 3: Adapting
$t \rightarrow t + 1$

Figure 3.3: Flow diagram that shows the adaptation of ACS estimator. The target distribution are shown by solid lines, while the sampling distributions are plotted with dashed lines. The initial sampling distribution gradually tilts toward target distribution by reweighting sample proposals.

incremental importance weight $w_{i,t} = \frac{\pi(x)}{g^{(t-1)}(x)}$, which quantifies the discrepancy between target distribution $\pi(x)$ and current sampling distribution $g^{(t-1)}(x)$. Then we average out all the sample proposals $q_i^{(t)}(x)$ based on their $w_{i,t}$ and generate new sampling distribution $g^{(t)}(x)$ for next iteration. This concept is very similar to the methodology in Kernel Density Estimation (KDE) method, where each $q_i^{(t)}(x)$ represents a kernel. As iteration continues, more observations will be added to target distribution $\pi(x)$ to make it closer to real failure region distribution. And our sampling distribution approximates the target distribution $\pi(x)$ through a random walk, as shown in Figure 3.3. With this ACS adaptation scheme, our sampling distribution can search to achieve full failure region coverage, and focus on the most important failure boundaries.

3.3.3 Comparison with other IS estimators

In comparison with conventional static IS with deterministic mixture, our ACS algorithm has two primary advantages. First of all, our ACS estimator can spread out throughout the parametric space while maintaining sample diversity. To be specific, a major challenge for other AIS methodologies in statistics community is the diversity of samples is prone to degenerate as iteration proceeds. These approaches utilize a resampling scheme to sample with replication from a series of sample proposals. Resampling method tilts to approximate failure event distribution. However, this random walk is generated globally in a greedy fashion, which may focus on the major failure regions and neglect the minor ones. And it also yields higher computational complexity. Our ACS estimator, on the contrary, is completely parallelized in different directions, and the evaluation of failure probability is also performed in parallel. It accelerates the convergence of our estimation and all the failure regions on different directions are separately protected between iterations.

Moreover, ACS estimator outperforms other IS based estimator in terms of Effective Sample Size (ESS). ESS is defined as:

$$ESS = \frac{1}{\sum_{i=1}^{M}(\bar{w}_{i,t})^2} \tag{3.7}$$

where $\bar{w}_{i,t}$ is the normalized incremental importance weight for each sample in all iterations. It reflects the number of samples that contribute to corresponding estimator. Our ACS estimator bias the sampling distribution to the samples with larger $\bar{w}_{i,t}$. In the next iteration, more samples with larger weight will be generated. Thus $\bar{w}_{i,t}$ is naturally balanced and it is more likely to converge to optimal $w^* = \frac{1}{M}$, which maximizes ESS to M. In this case, each sample uniformly contributes to the estimator and the estimator is stabilized.

3.4 Experiment Result

In this section, we first evaluate our proposed ACS method on a typical SRAM bit cell with 18 variables. More realistically, we verify ACS on high-dimensional SRAM column with 576 variables. We also implement different methods, including MC, HSCS [WBH16] and

Table 3.1: Accuracy and efficiency comparison on 18-dimensional SRAM bit cell

	MC	HSCS	AIS	Proposed
Failure prob.(error)	1.24e-5(0%)	1.18e-5(4.6%)	1.32e-5(6.1%)	1.22e-5(1.5%)
Presampling # sim.	0	7000	3000	1100
Importance Sampling # sim.	1e7	8688	5111	1736
Total # sim. (speedup)	1e7(1X)	15688(637X)	8111(1233X)	2836(3526X)

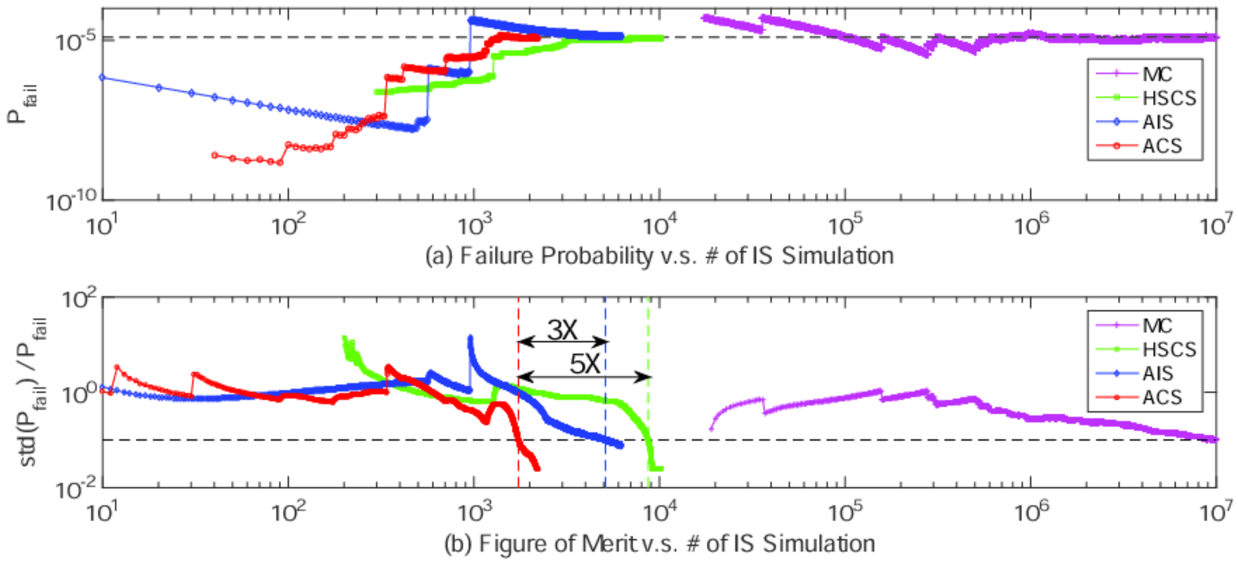


Figure 3.4: Evolution comparison of failure prob. and FOM on SRAM bit cell

AIS [SLYH18], and compare from both accuracy and efficiency perspective. The experiment environment is HSPICE with SMIC 40nm model.

3.4.1 Experiments on 6T SRAM Bit Cell

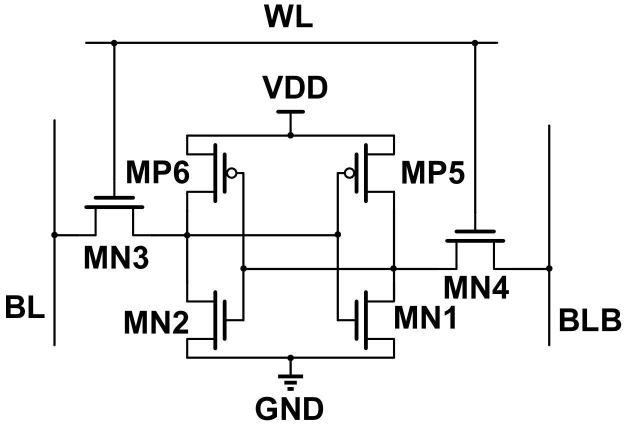


Figure 3.5: The schematic of typical 6T SRAM cell

Figure 3.5 shows the schematic of a typical 6-transistors SRAM bit cell. Four transistors $MN1$, $MP5$, $MN2$ and $MP6$ form two cross-coupled inventers and utilize two steady states '0' and '1' to store data in the memory cell. The other two access transistors MN3 and MN4 work as switches for read and write operation. In this experiment, we consider various failure mechanisms, including reading failure, writing failure and data retention failure. We will compare different methods (MC, HSCS, AIS, proposed) in terms of accuracy and efficiency.

3.4.1.1 Accuracy Comparison

To verify the accuracy of proposed ACS algorithm, Figure of Merit (FOM), ρ, is applied to represent the accuracy convergence and confidence of estimation. The definition of FOM is:

$$\rho = \frac{\sqrt{\sigma^2_{\hat{P}_{fail}}}}{\hat{P}_{fail}} \tag{3.8}$$

where $\hat{P}_{fail}$ indicates failure probability and $\sigma_{\hat{P}_{fail}}$ indicates the standard deviation of $\hat{P}_{fail}$. We define one estimation has $(1-\epsilon)100\%$ accuracy with $(1-\delta)100\%$ confidence when

$\rho < \epsilon\sqrt{log(1/\delta)}$. In our experiment, we draw a dashed line when ρ reaches 0.1 to indicate the 90% accuracy with 90% confidence, which is an extensively used ρ value in the literature [WBH16, SLYH18].

We compare the convergence of failure probability estimation and FOM calculation in Figure 3.4. We observe that the estimation of MC, HSCS, AIS and proposed ACS all converge when sufficient simulations are allowed. As illustrated in Table 3.1, the ground truth MC estimation is 1.24e-5 (4.37σ). Among these methods, the proposed ACS algorithm is most accurate with only 1.5% relative error, while the results of HSCS and AIS have 4.6% and 6.1% relative error, respectively.

3.4.1.2 Efficiency Comparison

The efficiency of MC, HSCS, AIS and ACS is shown in Figure 3.4. We notice that ACS has the fastest convergence among these algorithms. It is attributed to our unbiased estimator, which is updated parallelly in disjoint clusters. It is far more efficient than static sampling distribution in HSCS and global resampling scheme in AIS. To be specific, as shown in Table 3.1, our ACS method converge to 90% confidence with only 1736 IS simulations, while HSCS and AIS need 8688 and 5111 IS simulations to obtain the same FOM value. In addition, the proposed ACS algorithm requires 1100 presampling simulations. In comparison, HSCS and AIS need 7000 and 3000 times, respectively. This result demonstrates that proposed method is less sensitive to initial states. In total, ACS algorithm can achieve 3526X speedup over MC, 5X over HSCS and 3X over AIS.

3.4.2 Experiments on SRAM Column Circuit

A simplified schematic of SRAM column consisted of 32 bit cells is shown in Figure 3.6. Compared with a single bit cell in the previous low-dimensional experiment, we consider the impact of peripheral circuit and generate a more accurate estimation of failure probability. The configuration in Figure 3.6 demonstrates the worst-case scenario of read operation, in which accessed bit $CELL < 0 >$ stores “0” and other idle bits store “1”. In this case, the leakage current through idle bits (storing complementary value to the accessed bit) increases read access time and impedes a successful read. In our experiment, we simulate various SRAM failures in reading, writing and standby mode. There are in total 576 variation parameters in this test case, which is a high-dimensional problem.

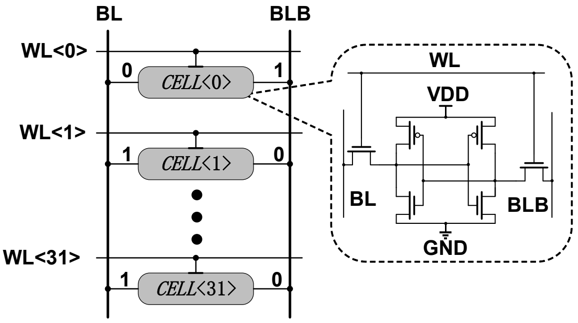


Figure 3.6: The schematic of 576-dimensional SRAM column circuit

3.4.2.1 Comparison of Visualized Failure Regions

In order to investigate the different performance in high-dimensional circuit case, we project the sample points on two most important dimensions, $Vth0$ of $MN1$ and $Vth0$ of $MN3$. Figure 3.7 shows the visualized multiple failure regions in 2D parametric space for different methods. Based on MC sampling in Figure 3.7(a), three disjoint failure regions are clearly displayed. We notice that it contains two major failure regions and a minor one in the lower right corner. Here green dot denotes the mean value for original distribution on these two dimensions.

As shown in Figure 3.7(b), HSCS groups failure samples into clusters, and utilizes min-norm points to generate shifted Gaussian distributions that cover multiple failure regions. The location of min-norm points are depicted as black triangles in Figure 3.7(b). It is, however, less effective on high-dimensional circuit, as it is challenging to locate the min-norm points on the failure boundary. With biased min-norm points, we note that majority of captured samples are not generated in failure regions. Thus it requires more simulations in both presampling and IS steps. In Figure 3.7(c), AIS converges to wrong P_{fail} value because only two failure regions are notified. The minor failure region in the lower right corner disappears during iterations. It is due to the unbalanced weights for each sample. Samples with smaller weights are more prone to be eliminated in the global resampling procedure. The proposed ACS algorithm successfully searches for all three failure regions after a few iterations. And we notice that the whole sample set is focused on the most important failure boundary, which dominates the failure probability estimation. Therefore, ACS method outperforms other methods in terms of failure region exploration and convergence speed.

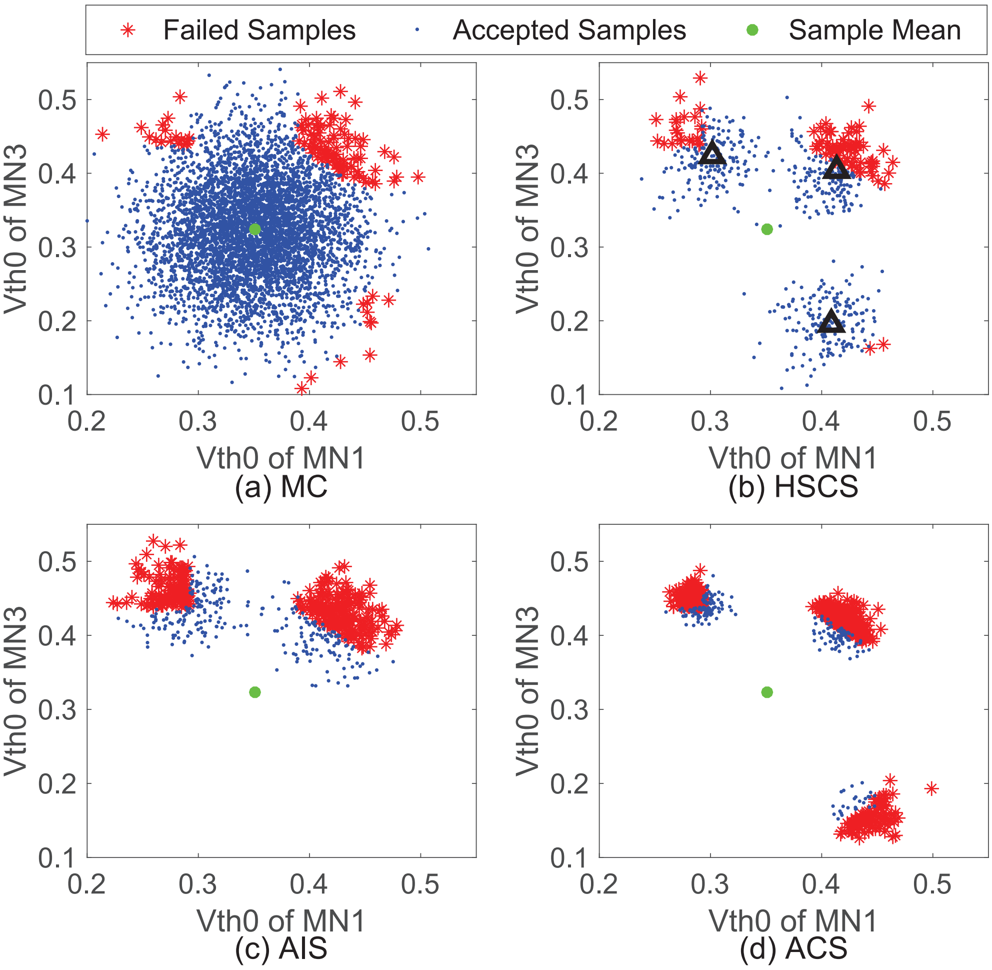


Figure 3.7: Multiple failure region coverage test (failed samples/ accepted samples/ sample mean are colored)

Table 3.2: Accuracy and efficiency comparison on 576-dimensional SRAM column

	Monte Carlo	HSCS	AIS	Proposed
Failure prob.(error)	1.60e-5(0%)	9.82e-6(error)	2.23e-6(error)	1.55e-5(3.1%)
Presampling # sim.	0	18000	5000	2000
Importance Sampling # sim.	1e7	28699	11253	2878
Total # sim. (speedup)	1e7(1X)	46699	16253	4878(2050X)

3.4.2.2 Accuracy and Efficiency Comparison

In this section, we compare the evolution of the failure probability and FOM in Figure 3.8. The ground truth failure probability is estimated by brute-force MC. Among HSCS, AIS and ACS algorithms, only our ACS method is capable of converging to gold failure probability. HSCS converges to wrong failure rate with much slower speed because static deterministic mixture IS estimator cannot work in high-dimensional case. AIS also cannot generate correct failure rate estimation. For multi-failure-region SRAM column circuit, some less important failure regions are neglected by AIS global resampling process, which leads to smaller failure probability.

To be specific, as shown in Table 3.2, the gold failure probability estimated by MC is 1.6e-5 (4.3σ) with 10 million simulations. Only ACS algorithm succeeds in giving accurate estimation with 3.1% relative error. The number of simulations in presampling and IS steps are 2000 and 2878, respectively. Therefore, the proposed ACS algorithm can obtain 2050X speedup w.r.t MC.

3.5 Conclusion

In this chapter, we present an Adaptive Clustering and Sampling method to efficiently estimate the rare-event failure probability of SRAM circuits. This method first applies hyperspherical presampling to generate a set of initial samples in high dimension. Next, an iterative clustering and sampling scheme is performed to search for failure regions and update

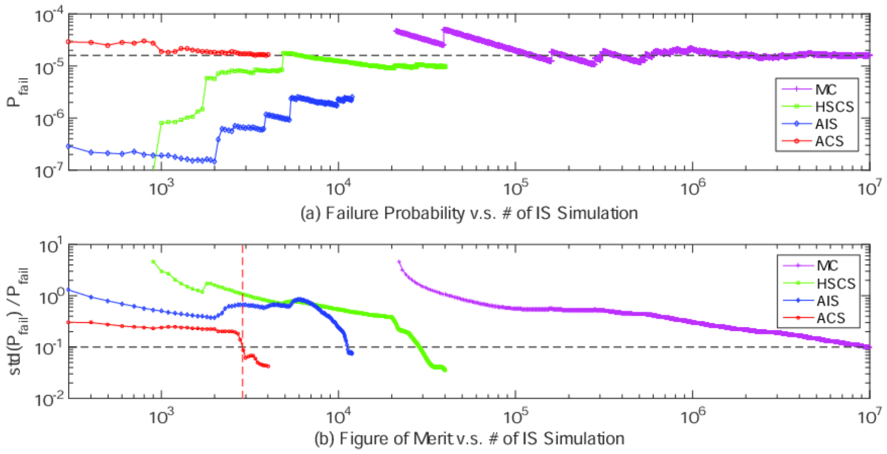


Figure 3.8: Evolution comparison of failure prob. and FOM on SRAM Column

estimation. The experiments demonstrate that proposed algorithm can provide extremely high accuracy and efficiency. Experiments on SRAM bit cell indicate that ACS achieves 3526X speedup over MC and 3-5X over other state-of-the-art methods. On SRAM column circuit in high dimension and with multiple failure regions, ACS is 2050X faster than MC method, while other IS based approaches fail to provide reasonable accuracy.

CHAPTER 4

Meta-Model based High-Dimensional Yield Analysis using Low-Rank Tensor Approximation

4.1 Summary

In this chapter, we propose a novel and efficient polynomial-based meta-model with tensor structure to tackle the challenging high-dimensional yield analysis problem. The specific contributions include:

- Derivation and formulation our meta-model in tensor spaces. The proposed meta-model is constructed with canonical decomposition. It represents a multi-way tensor in high-dimensional space into a finite sum of rank-one tensors. As tensor rank is independent of circuit dimension, our model successfully bridges the gap between circuit complexity and model scalability.
- An efficient yet effective greedy solver with sparse constraints. It is a constructive algorithm to successively minimize along tensor rank, thus heuristically converge to optimal solution. Moreover, the sparsity induced by circuit dimension is treated by considering regularized problems.
- An adaptive sampling scheme to reduce circuit simulation. We separate the whole sampling procedure into a series of incremental sampling iterations. With carefully designed sample weights, our sampling strategy is able to collect more informative ones around failure boundaries.

4.2 Polynomial-based Meta-Modeling

When P_{fail} is an extremely small value, standard MC becomes inefficient because it requires millions of simulations to capture one single failure event. In order to reduce computational cost, we attempt to construct an efficient meta-model using relatively small amount of SPICE simulations with reasonable budget. This mapping between d-dimensional input variable and meta-model response can be written as:

$$\boldsymbol{X} \in \mathbb{R}^d \mapsto \mathcal{M}(\boldsymbol{X}) \in \mathbb{R} \tag{4.1}$$

Figure 4.1 shows an illustrative global meta-modeling process with 2-dimensional input variable. In order to obtain an accurate meta-model, we need to choose appropriate model structure $\mathcal{M}$ and sampling strategy. In the yield analysis scenario, samples located near failure region boundaries, which separate the "0", "1" values of indicator function $I(X)$, are of more interest.

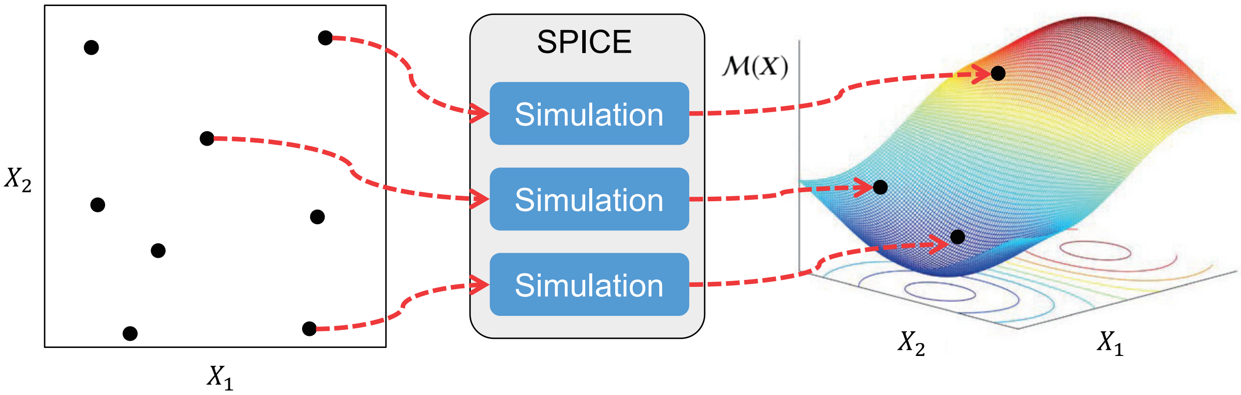


Figure 4.1: The construction process of meta-model which maps between input variable and output performance

Our proposed work focuses on the category of polynomial-based meta-model, because of its flexibility to model various functions and simplicity to implement. For example, Polynomial Chaos Expansions (PCE) [Sud08] has been extensively used in the context of uncertainty quantification. The key concept of PCE is to expand the model response onto a series of orthonormal polynomial basis along each dimension:

$$\hat{Y} = \mathcal{M}_{PCE}(\boldsymbol{X}) = \prod_{i=1}^{d} \left(\sum_{k=0}^{n_i} \alpha_k^{(i)} \phi_k^{(i)} \right) \tag{4.2}$$

However, PCE suffers from the problem of "curse of dimensionality". We note that the number of unknown coefficients in Equation (4.2) is $\prod_{i=1}^{d}(n_i + 1)$, which increases exponentially with input dimension d. Generally, the number of required simulations is 2-3 times the number of unknowns, which is infeasible for high-dimensional circuit cases. Alternatively, we exploit the tensor-product structure of the polynomial basis, which can reduce the number of coefficients by order of magnitude.

4.3 Low-rank Tensor Approximation Formulation

In order to construct a non-intrusive meta-model $Y = \mathcal{M}(\boldsymbol{X})$ described in Section 2.2, our LRTA method formulates it into a highly-compressed tensor format. We first introduce some basic definitions of tensor subsets. The high-dimensional tensor space can be represented as $\mathcal{S} = \mathcal{S}^1 \otimes \cdots \otimes \mathcal{S}^d$. In order to model circuit performance metric Y in space $\mathcal{S}$, LRTA considers a sequence of approximations in rank-one tensor subset $\mathcal{T}_1 \subset \mathcal{S}$, which is defined as:

$$\mathcal{T}_1 = \left\{ v(\boldsymbol{X}) = \left(\otimes_{i=1}^{d} w^{(i)}\right)(\boldsymbol{X}) = \prod_{i=1}^{d} w^{(i)}(X_i); w^{(i)} \in \mathcal{S}^i \right\} \tag{4.3}$$

where $\boldsymbol{X}$ is a d-dimensional multivariate circuit variable, and $w^{(i)}$ is a univariate function of X_i.

We note that tensor space has the property that $\mathcal{S} = span(\mathcal{T}_1)$, such that each element in $\mathcal{S}$ can be expressed as a linear combination of rank-one tensors $v_l(\boldsymbol{X})$, as shown in Equation (4.4):

$$\mathcal{S} = \left\{ Y = \sum_{l \in I_n} b_l v_l(\boldsymbol{X}); \quad v_l \in \mathcal{T}_1, b_l \in \mathbb{R} \right\} \tag{4.4}$$

where b_l denotes the l-th normalization constant for the corresponding rank-one component.

In practice, we truncate this canonical decomposition expression with small R and sufficient accuracy. Such decomposition is thereby named as *Low-Rank Tensor Approximations*:

$$\hat{Y}^R = \mathcal{M}^R(\boldsymbol{X}) = \sum_{l=1}^{R} b_l v_l(\boldsymbol{X}); \quad v_l \in \mathcal{T}_1, b_l \in \mathbb{R} \tag{4.5}$$

Generally, the optimal tensor rank R is not known as *priori*. An appropriate parameter tuning procedure will be later discussed in Section 4.3.

Next, we expand each rank-one tensor $v_l(\boldsymbol{X})$ to a polynomial representation

$$v_l(\boldsymbol{X}) = \prod_{i=1}^{d} w_l^{(i)}(X_i) = \prod_{i=1}^{d} \left(\sum_{k=0}^{n_i} z_{k,l}^{(i)} \phi_k^{(i)}(X_i) \right) \tag{4.6}$$

where $\boldsymbol{\phi}^{(i)}$ is a set of orthonormal polynomial basis function for the i-th circuit variable, $\{z_{k,l}^{(i)}\}$ is the corresponding set of coefficient to be solved, and n_i is the maximum degree of polynomial expansion. Intuitively, properly chosen polynomial basis families will accelerate the convergence of our LRTA method. In this paper, we utilize Hermite polynomial basis because each circuit variable is modeled as Gaussian variable [XK02]. Another excellent insight is that the number of unknown coefficients of each rank-one tensor $v_l(\boldsymbol{X})$ is $\sum_{i=1}^{d}(n_i + 1)$, which grows linearly with dimension d. This property makes LRTA particularly promising for dealing with high-dimensional circuit problems.

By substituting Equation (4.6) into Equation (4.5), our proposed LRTA meta-model is formulated as:

$$\hat{Y} = \mathcal{M}^R(\boldsymbol{X}) = \sum_{l=1}^{R} b_l \left(\prod_{i=1}^{d} \left(\sum_{k=0}^{n_i} z_{k,l}^{(i)} \phi_k^{(i)}(X_i) \right) \right) \tag{4.7}$$

Compared with the conventional PCE format described in Section 2.2, our LRTA construction partitions a single large-size minimization problem with a sequence of smaller ones. However, solving for $\{b_l\}$ and $\{z_{k,l}^{(i)}\}$ is non-trivial. In the next section, we propose a novel adaptive solver with a greedy scheme and sparsity constraints.

The overall algorithm of LRTA yield analysis method is summarized in Algorithm 1.

Algorithm 5: Framework of LRTA Yield Analysis Method

```
Input: Maximum tensor rank $R$,
       Maximum polynomial degree $n_i$
Output: Failure probability estimation $\hat{P}_{fail}$
1. Construct LRTA meta-model
       $\hat{Y} = \mathcal{M}^R(\boldsymbol{X}) = \sum_{l=1}^{R} b_l \left( \prod_{i=1}^{d} \left( \sum_{k=0}^{n_i} z_{k,l}^{(i)} \phi_k^{(i)}(X_i) \right) \right)$
2. Use Proposed Adaptive Solver to compute coefficients $\{b_l\}$ and $\{z_{k,l}^{(i)}\}$
3. Use LRTA Meta-model to compute failure probability
       $\hat{P}_{fail} = \frac{1}{N} \sum_{i=1}^{N} I\left(\mathcal{M}^R(\boldsymbol{X}) \in S\right)$
```

4.4 Proposed Adaptive Solver

4.4.1 Implementation with Greedy Algorithm

In our LRTA algorithm, the meta-model response $\mathcal{M}^R(\boldsymbol{X})$ is constructed from heuristically accumulating R rank-one tensor components. Our greedy algorithm consists of two major steps. In the correction step, at certain iteration l, rank-one tensor v_l attempts to minimize the mismatch between current meta-model response $\mathcal{M}^{l-1}(\boldsymbol{X})$ and realistic observations Y. For each v_l, the polynomial coefficients $\{z_l\}$ are determined by an Alternating Minimization approach. In the following normalization step, we update the entire set of previous normalization coefficients $\{b_1, \ldots, b_l\}$ by solving a minimization problem. Our algorithm iterates between these steps until pre-defined rank R is reached. Implementation details are as follows.

Correction step. Let $\mathcal{R}^{l-1}(\boldsymbol{X})$ denote the approximation residual after the $(l-1)$-th iteration:

$$\mathcal{R}^{l-1}(\boldsymbol{X}) = Y - \mathcal{M}^{l-1}(\boldsymbol{X}) \tag{4.8}$$

In the next iteration, a new rank-one tensor v_l works as a correction function which approximates the residual:

$$v_l = \operatorname*{argmin}_{\gamma \in \mathcal{T}_1} \|\mathcal{R}^{l-1} - \gamma\|^2 \tag{4.9}$$

Here the solution of Equation (4.9) is calculated by an Alternating Minimization approach. Along each dimension $i \in \{1, ..., d\}$, we introduce a series of minimization problems to solve for $z_l^{(j)}$, while coefficients on other dimensions remain unchanged. To be specific, each polynomial coefficient $z_l^{(j)}$ is determined as:

$$z_l^{(j)} = \operatorname*{argmin}_{\boldsymbol{\zeta} \in \mathbb{R}^{n_j+1}} \left\| \mathcal{R}^{l-1} - \left(\prod_{i \neq j} w_l^{(i)} \right) \left(\sum_{k=0}^{n^j} \zeta_k^{(j)} \phi_k^{(j)} \right) \right\|^2 \tag{4.10}$$

Equation (4.10) has a classical form of minimization problem with $(n_j + 1)$ unknowns, which can be solved directly using Least Square method.

Normalization step. Once a new rank-one correction v_l is constructed at the l-th iteration, we need to calibrate all the previous normalization constant $\boldsymbol{b} = \{b_1, \ldots, b_l\}$. It is computed by solving a series of minimization problem:

$$\boldsymbol{b} = \operatorname*{argmin}_{\beta \in \mathbb{R}^l} \left\| Y - \sum_{m=1}^{l} \beta_m v_m \right\|^2 \tag{4.11}$$

As iteration continues, the size of vector $\boldsymbol{b}$ increases along with iteration lable l. We also notice that b_l can represent the significance of tensor component. If particular b_l is negligible, corresponding tensor component v_l can be discarded without sacrificing accuracy. This procedure further decreases tensor rank R and thus reduces model complexity.

4.4.2 Sparse Constraint

In realistic high-dimensional yield analysis application, the contribution of different circuit variables $\{X_i\}_{i=1}^d$ w.r.t. performance metric Y varies drastically. The majority of circuit variables only have weak influence on performance. Thus polynomial coefficient set $\{z_l\}$ contains large amount of zero elements or elements with extremely small magnitude. As a consequence, Equation (4.10) can be improved by adding a ℓ_1-norm constraint $\|\boldsymbol{\zeta}\|_1 \leq \delta$. As ℓ_1-norm is strictly convex, we can equivalently rewrite it as a generalized LASSO [Tib96] problem:

$$z_l^{(j)} = \operatorname*{argmin}_{\boldsymbol{\zeta} \in \mathbb{R}^{n_j+1}} \left\| \mathcal{R}^{l-1} - \left(\prod_{i \neq j} w_l^{(i)} \right) \left(\sum_{k=0}^{n^j} \zeta_k^{(j)} \phi_k^{(j)} \right) \right\|^2 + \lambda \|\boldsymbol{\zeta}\|_1 \tag{4.12}$$

which is a convex optimization problem. Here λ is a suitable regularization factor. In our implementation, problem (4.12) is solved by least angle regression algorithm [EHJ$^+$04], which is a commonly used stagewise procedure for accelerating LASSO problems.

4.4.3 Generic Cross Validation for Parameter Tuning

In our LRTA meta-model formulated with polynomial basis, the tensor rank R and polynomial degree n_i are pre-defined. Intuitively, the selection of parameter pair $\{R, n_i\}$ exhibits a trade-off between approximation accuracy and model complexity. In this section, we propose a 3-fold Cross Validation (CV) scheme to explore parametric space and select optimal parameter pair. The overall procedure is demonstrated as follows:

- Partition whole sample set $\mathcal{Q}$ into three test sets with equivalent samples $\mathcal{U}_i$, $i = \{1, 2, 3\}$. Then training sets consist of corresponding remaining samples $\mathcal{V}_i = \mathcal{Q} n \mathcal{U}_i$, $i = \{1, 2, 3\}$.

- Execute our adaptive solver on each training set $\mathcal{V}_i$ with tuned parameters ranging from $1 \leq R \leq R_{max}$ and $1 \leq n_i \leq n_{max}$. For each parameter pair $\{R, n_i\}$, we compute their mean square errors $\bar{\varepsilon}_{R,n_i}$ comparing with the evaluations from the test sets $\mathcal{U}_i$. Optimal parameter pair $\{R, n_i\}_{op}$ is thereby selected with minimum $\bar{\varepsilon}_{R,n_i}$.

- Execute adaptive solver on the whole sample set $\mathcal{Q}$ with optimal $\{R, n_i\}_{op}$.

4.4.4 Improvement via Adaptive Sampling

Classical deterministic sampling approaches generate random samples over the entire parametric space. However, it is more desirable to capture more informative samples which induce larger prediction error or locate near failure boundaries. The performance of proposed solver can be improved if we utilize "important" samples to set up the rank-R tensor.

Here we partition the whole sampling procedure into T incremental sampling processes. It starts from an initial sample set $\mathcal{Q}$, and a new sample set $\mathcal{Q}_c$ is sequentially added in each iteration by our adaptive sampling framework, which is summarized in Algorithm 2.

Algorithm 6: Adaptive Sampling Algorithm

```
Initialization: Generate $M$-element initial sample set $\mathcal{Q}$ from the multivariate
 PDF of process variations $p(\boldsymbol{X})$
for $t = 1, 2, ..., T$ do
    1. Metamodeling:
        Construct $t$-th iteration LRTA meta-model $\mathcal{M}^{(t)}(\boldsymbol{X})$
        with current sample set $\mathcal{Q}^{(t)}$
    2. Weighting:
        (a) For each sample $\boldsymbol{X}_i^{(t)}$ in $\mathcal{Q}^{(t)}$, generate a discrete
            Gaussian sampling distribution $N_i^{(t)}$
        (b) Assign weight function $w_i^{(t)}$ to each distribution $N_i^{(t)}$
                $w_i^{(t)} = \left|Y - \mathcal{M}^{(t)}(\boldsymbol{X}_i)\right| \cdot p(\boldsymbol{X}_i)$
    3. Sampling:
        (a) Average out $N_i^{(t)}$ based on $w_i^{(t)}$ to construct new
            sampling distribution $G^{(t)} = 1/\left(\sum w_i^{(t)}\right) \cdot \sum w_i^{(t)} N_i^{(t)}$
        (b) Generate $M_c$ new samples $\mathcal{Q}_c^{(t)}$ from distribution $G^{(t)}$
        (c) Update sample set $\mathcal{Q}^{(t+1)} = \mathcal{Q}^{(t)} \cup \mathcal{Q}_c^{(t)}$
end
```

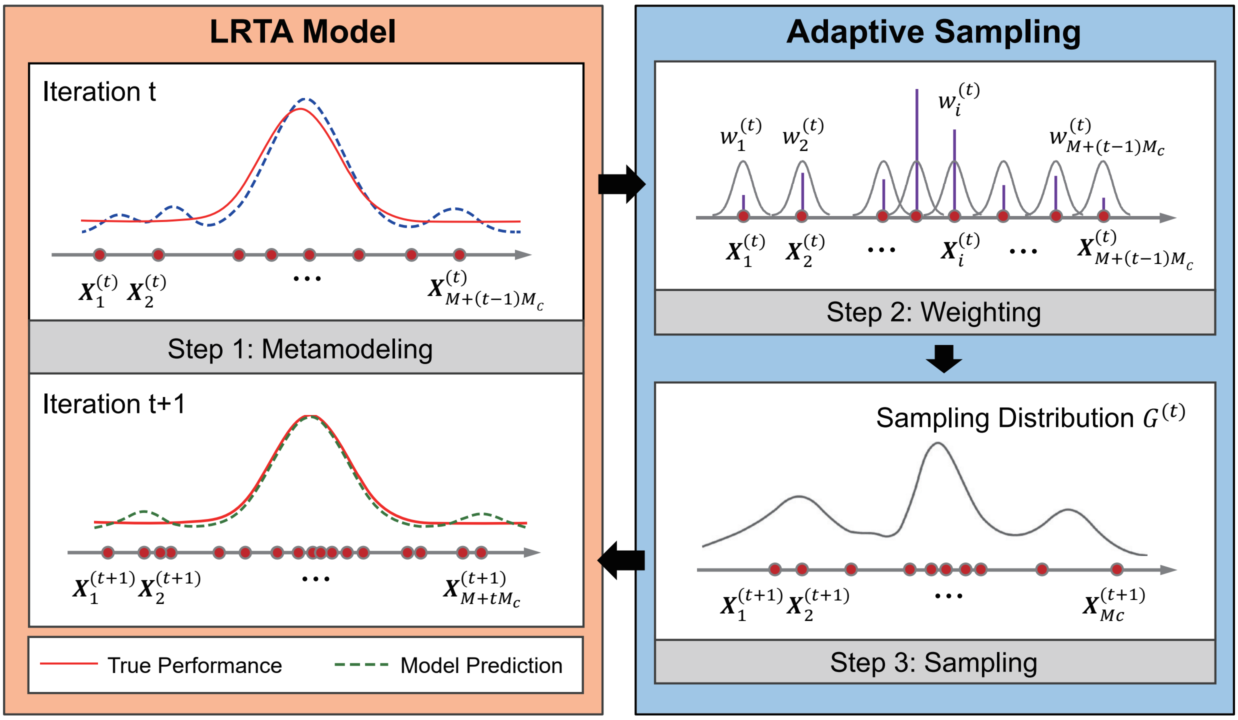

LRTA Model
Iteration t
$X_1^{(t)}$ $X_2^{(t)}$ ... $X_{M+(t-1)M_c}^{(t)}$
Step 1: Metamodeling
Iteration t+1
$X_1^{(t+1)}$ $X_2^{(t+1)}$ ... $X_{M+tM_c}^{(t+1)}$
True Performance
Model Prediction
Adaptive Sampling
$w_1^{(t)}$ $w_2^{(t)}$ $w_i^{(t)}$ $w_{M+(t-1)M_c}^{(t)}$
$X_1^{(t)}$ $X_2^{(t)}$ ... $X_i^{(t)}$ ... $X_{M+(t-1)M_c}^{(t)}$
Step 2: Weighting
Sampling Distribution $G^{(t)}$
$X_1^{(t+1)}$ $X_2^{(t+1)}$ ... $X_{Mc}^{(t+1)}$
Step 3: Sampling

Figure 4.2: Flow diagram of adaptive sampling scheme. The LRTA model is calibrated with multiple iterations of sampling distribution adjustments. Our sampling strategy adaptively reweights past samples and tends to focus on distributions with higher weight.

Figure 4.2 is an illustrative flow diagram that demonstrates our adaptive sampling scheme. The concept is very similar to the methodology in Kernel Density Estimation (KDE), where each $N_i^{(t)}$ represents a kernel. As iteration proceeds, new samples are prone to locate around kernels with larger weight w. Here w is a fused metric, whose first term quantifies meta-model accuracy and second term evaluates sample importance. In practice, after a few iterations, our sampling distribution will tilt toward the failure region boundaries, thus improve the prediction quality.

4.5 Experimental Results

The proposed LRTA method is first evaluated on a typical SRAM bit cell with 18 variables. More realistically, we validate our LRTA on a high-dimensional SRAM column with peripheral circuits, which has in total 597 variables. We also implemented MC as ground truth, and obtained codes for Hyperspherical Clustering and Sampling (HSCS) [WBH16] and Adaptive Importance Sampling (AIS) [SLYH18] for comparison. The experiment environment is HSPICE with SMIC 40nm model.

4.5.1 Experiments on 6T SRAM Bit Cell

Figure 4.3 shows the schematic of typical 6T SRAM bit cell. Four transistors form two cross-coupled inverters and use two steady states to store data in the cell. The other two transistors control accessing to the storage cell during read and write operations. In our experiments, we simulate various types of SRAM failures in reading, writing and standby mode. We evaluate different methods (MC, HSCS, AIS, proposed) to compare accuracy and efficiency.

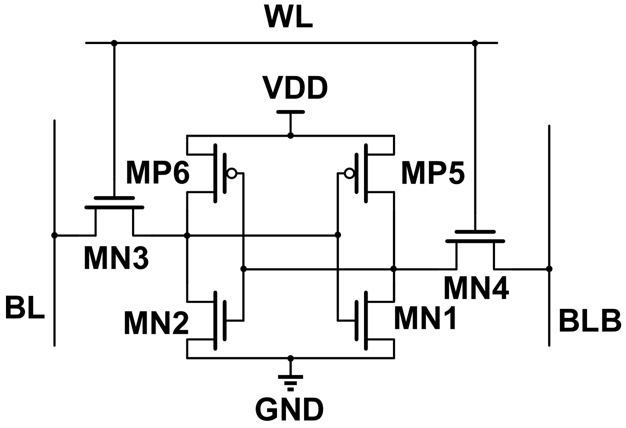


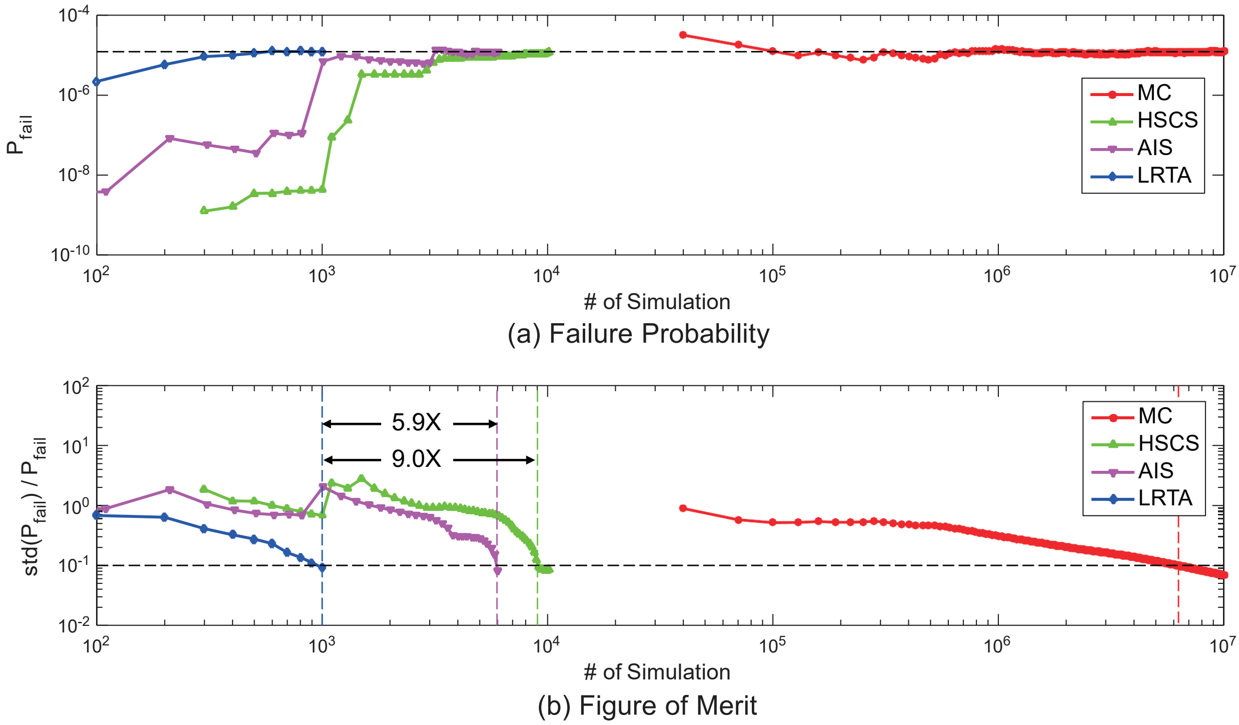
Figure 4.3: The schematic of typical 6T SRAM cell

4.5.1.1 Accuracy Comparison

In order to verify the accuracy of proposed LRTA method, we introduce Figure of Merit ρ to characterize the accuracy convergence and confidence of estimation, which is defined as:

$$\rho = \frac{\sqrt{\sigma^2_{\hat{P}_{fail}}}}{\hat{P}_{fail}} \tag{4.13}$$

where $\hat{P}_{fail}$ is the estimation of P_{fail} and $\sigma_{\hat{P}_{fail}}$ denotes its standard deviation. With this definition, we can declare one estimation has $(1-\epsilon)\times 100\%$ accuracy with $(1-\delta)\times 100\%$ confidence when $\rho < \epsilon\sqrt{log(1/\delta)}$. In our experiments, the dashed line indicates ρ reaches 0.1, which represents estimation reaches a steady state with 90% confidence. The estimated failure probability is thereby defined as the corresponding 90% confidence stable value.



(a) Failure Probability

(b) Figure of Merit

Figure 4.4: Evolution comparison of failure prob. and FOM on 18-dimensional SRAM bit cell

Table 4.1: Accuracy and efficiency comparison on 18-dimensional SRAM bit cell

	MC	HSCS	AIS	Proposed
Failure prob.	1.24e-5	1.18e-5	1.32e-5	1.27e-5
Relative error	golden	4.8%	6.4%	2.4%
# Sim. runs	6.3e6	9019	5962	1000
Speedup	1X	698X	1056X	6300X

Figure 4.4(a) demonstrates the evolution of failure probability estimation. We first notice

that the failure rate estimations from HSCS, AIS and proposed LRTA all converge to ground-truth MC value when sufficient simulations are allowed. As shown in Table 4.1, ground-truth MC estimation is 1.24e-5 (4.37σ). Our proposed LRTA method is the most accurate one with only 2.4% relative error, and HSCS and AIS approaches have 4.8% and 6.4% relative error, respectively.

4.5.1.2 Efficiency Comparison

Then we compare different methods in terms of efficiency. Evolution of P_{fail} convergence and FOM evaluation are plotted in Figure 4.4, the following observations can be made:

- First, sampling-based methods, such as HSCS and AIS, are highly sensitive to the sampling distribution. We can observe that the P_{fail} estimation curve changes abruptly as discrete failure sample is added to the sample set. In contrast, our LRTA method performs a global approximation, which can provide very consistent estimation as sample set grows.
- Second, the FOM curve of our LRTA method is monotonically decreasing, while HSCS and AIS fluctuate before asymptotically reaching 90% confidence. It is attributed to better stability of our estimation. This feature accelerates our estimation procedure and exhibits better convergence property.
- From Table 4.1, our LRTA method is capable of achieving 90% confidence estimation by using 1000 samples to construct an effective meta-model, while MC requires 6.3 million to generate golden estimation. In comparison, HSCS and AIS need 9019 and 5962 simulations, respectively. As a conclusion, LRTA method can achieve 6300X speedup over MC, 9.0X over HSCS and 5.9X over AIS.

4.5.2 Experiments on SRAM Column Circuit

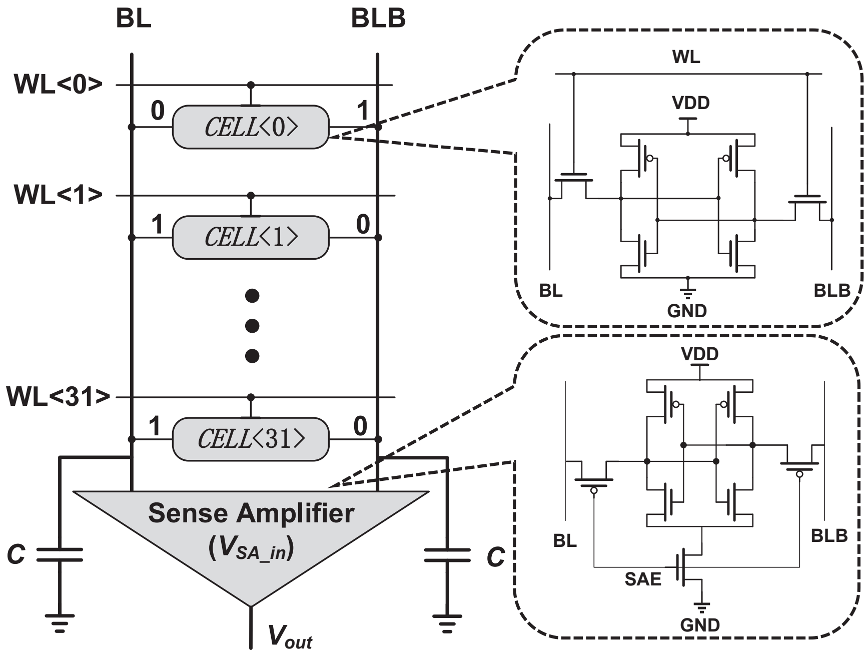
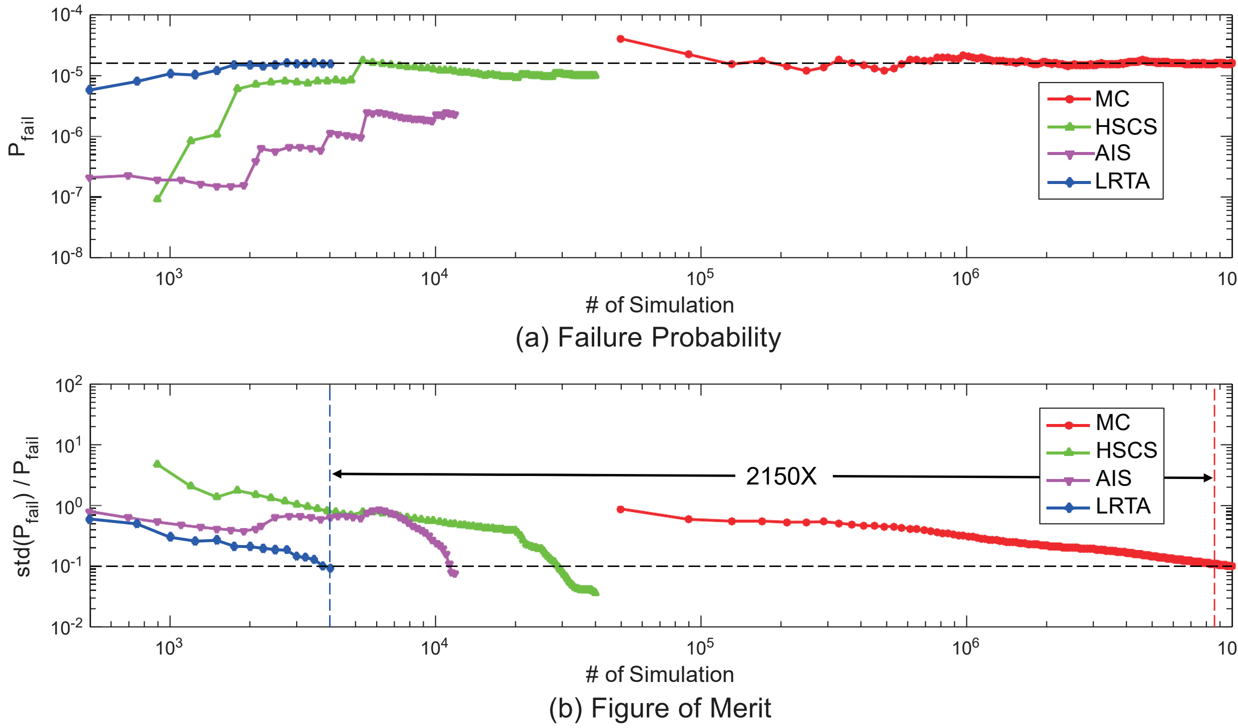


Figure 4.5: The schematic of 597-dimensional SRAM column with peripheral circuits

A simplified schematic of SRAM column circuit consisting of 32 bit cells and a sense amplifier is shown in Figure 4.5. There are 597 circuit variables in this case, which is a high-dimensional problem. Compared with the single bit cell low-dimensional setup in the previous section, we consider a more realistic scenario with the impact of peripheral circuits, thus generate a more accurate failure rate estimation. As an illustrative example, the worst case of read operation is configured in Figure 4.5, in which accessing bit $CELL$¡0¿ stores "0" and other idle bits store "1" without loss of generality. In this case, the leakage current through all the idle bits tends to increase read access time and impede successful read. Various failure mechanisms are considered in our experiments, including reading failure, writing failure and data retention failure.

4.5.2.1 Accuracy and Efficiency Comparison

In order to validate the accuracy and efficiency of LRTA method, we plot the evolution of P_{fail} and FOM in Figure 4.6. The ground truth MC requires 8.6 million simulations to generate confident estimation, which is 1.60e-5 (4.3σ). Among other algorithms, only our LRTA method is capable of converging to golden failure probability with 4.4% relative error. HSCS method converges to wrong failure rate with much slower speed because static deterministic Gaussian mixture cannot effectively cover failure regions in high-dimensional parametric space. The resampling procedure applied in AIS tends to neglect less important failure regions in high dimension, which leads to smaller P_{fail}. Contrasting to HSCS and AIS, LRTA method has much faster convergence speed by iteratively collecting important samples to calibrate our global meta-model. We observe that our LRTA method can achieve very promising estimation with 4000 simulations, which exhibits 2150X speedup w.r.t. MC.



(a) Failure Probability

(b) Figure of Merit

Figure 4.6: Evolution comparison of failure prob. and FOM on 597-dimensional SRAM column with peripheral circuits

4.5.2.2 Computational Complexity vs. Dimensionality

In this section, we investigate the relationship between computational cost and circuit dimension for different methods. Our experiment setting changes circuit scale by varying the number of bit cells in the SRAM column. It starts from one bit cell and a sense amplifier with 39 variables, then progressively increases the number of variables by sequentially adding bit cell to the column, up to 32 bit cells with 597 variables.

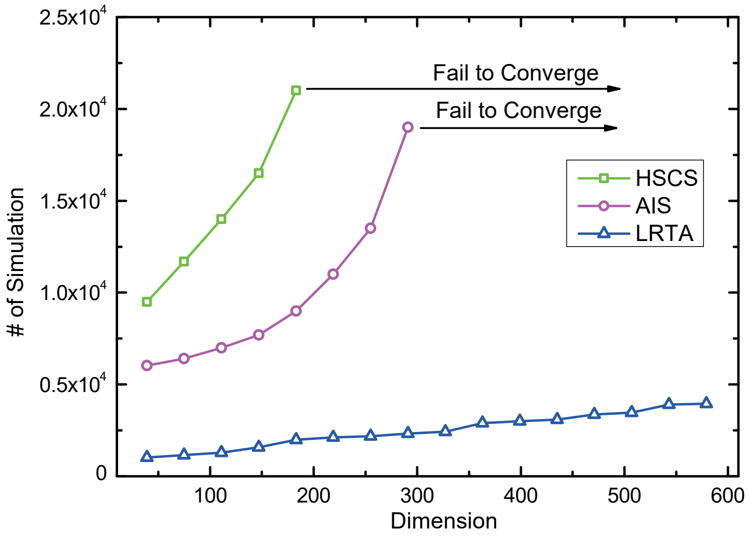


Figure 4.7: Comparison of required SPICE simulations versus circuit dimension for different methods. HSCS and AIS fail to converge to golden MC estimation as circuits scale up.

Figure 4.7 shows the comparison of simulation runs versus circuit dimension. It reveals that sampling based methods such as HSCS and AIS are less efficient and fail to converge because of "curse of dimensionality". On the contrary, the simulation cost of LRTA grows roughly linearly with the dimension, which is consistent with the number of polynomial basis in our compressed tensor expansion. We notice that proposed LRTA method becomes more competitive as dimension increases, and appears to be the only effective one when circuit dimension is larger than 300.

4.6 Conclusions

In this paper, we propose a meta-model based method using low-rank tensor approximation to efficiently estimate high-dimensional circuit failure probability. We first apply canonical tensor decomposition to formulate a LRTA meta-model. Next, we implement a robust solver using an efficient greedy algorithm with sparse constraints. To further reduce circuit simulations, an adaptive sampling framework is designed to select more informative samples and maintain reasonable estimation accuracy. The experimental results show that the proposed LRTA method can provide extremely high accuracy and efficiency. For SRAM bit cell with 18 variables, LRTA achieves 6300X speedup over MC and 6-9X over other state-of-the-art methods. For 597-dimensional SRAM column with peripheral circuits, LRTA is 2150X faster than MC method, while other approaches fail to converge to correct failure probability. Experiments also demonstrate that the simulation cost of proposed LRTA method increases linearly with circuit dimension, which is appealing for high-dimensional circuit problems.

CHAPTER 5

Meta-Model based Importance Sampling Algorithm

5.1 Motivation

Our motivation to design MIS algorithm is to preserve the strong points of other adaptive importance sampling algorithms (such as AIS[SLYH18] or PMC [CGMR04]), while exploiting the freedom given by meta-model to develop optimal sampling strategy. Our meta-model is initialized with Latin Hypercube Sampling (LHS) method, and updated iteratively. For each intermediate meta-model, we utilize it to approximate optimal sampling distribution, and enrich the sample set by performing Markov Chain Monte Carlo (MCMC) sampling. We construct intermediate IS estimators for each meta-model and use Deterministic Mixture (DM) approach to come up with the final global estimator. Compared with static IS, this DM strategy presents advantages in terms of stability and variance. Besides, no additional IS simulations are needed to estimate failure probability.

5.2 Gaussian Process assisted Density Approximation

In order to approximate the optimal IS sampling density $g^{opt}(\boldsymbol{x})$ described in Section II, our MIS method utilizes a series of iterative GP meta-models. We choose GP model due to its flexibility for various types of circuit responses and simplicity to implement. Another good property is that it has a built-in error estimator to evaluate the variance of GP predictions, which characterizes the accuracy of model fitting. In literature, GP models have been extensively applied as emulators or surrogates for black-box functions [SWMW89].

A typical GP meta-model is defined by a mean function $\mu(\boldsymbol{X})$ and a covariance matrix

$\boldsymbol{K}(\boldsymbol{X})$:

$$\mathcal{M}(\boldsymbol{X}) \sim \mathcal{GP}\left(\mu(\boldsymbol{X}), \boldsymbol{K}(\boldsymbol{X})\right) \tag{5.1}$$

Here $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ stands for a training set of size n, and $\boldsymbol{K}$ is a kernel matrix with expression of:

$$\boldsymbol{K} = \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_1, \boldsymbol{x}_n) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_n, \boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_n, \boldsymbol{x}_n) \end{bmatrix} \tag{5.2}$$

For a new sample point $\boldsymbol{x}_*$, GP outputs a prediction $[\boldsymbol{y}, y_*]$ in the form of a jointly normal distribution:

$$\begin{bmatrix} \boldsymbol{y} \\ y_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu}(\boldsymbol{X}) \\ \mu(\boldsymbol{x}_*) \end{bmatrix}, \begin{bmatrix} \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_\epsilon^2 \boldsymbol{I} & \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{x}_*) \\ \boldsymbol{K}(\boldsymbol{x}_*, \boldsymbol{X}) & \boldsymbol{K}(\boldsymbol{x}_*, \boldsymbol{x}_*) \end{bmatrix} \right) \tag{5.3}$$

The prediction can be further formulated as a Gaussian random variable:

$$P(y_*|\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{x}_*) = \mathcal{N}\left(\mu(\boldsymbol{x}_*), \sigma^2(\boldsymbol{x}_*)\right) \tag{5.4}$$

The predictive mean and variance are as follows:

$$\mu(\boldsymbol{x}_*) = \mu(\boldsymbol{x}_*) + \boldsymbol{K}(\boldsymbol{x}_*, \boldsymbol{X})(\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_\epsilon^2 \boldsymbol{I})^{-1}(\boldsymbol{y} - \mu(\boldsymbol{X})) \tag{5.5}$$

$$\sigma^2(\boldsymbol{x}_*) = \boldsymbol{K}(\boldsymbol{x}_*, \boldsymbol{x}_*) - \boldsymbol{K}(\boldsymbol{x}_*, \boldsymbol{X})(K(\boldsymbol{X}, \boldsymbol{X}) + \sigma_\epsilon^2 \boldsymbol{I})^{-1} \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{x}_*) \tag{5.6}$$

With aforementioned GP meta-model, our MIS algorithm is able to approximate optimal IS sampling density $g^{opt}(\boldsymbol{x})$. First, we apply GP to construct a probabilistic classification function $\pi(\boldsymbol{x})$, which assembles the indicator function $I(\boldsymbol{x})$. We note that the value of ideal indicator function is either 0 or 1. Our $\pi(\boldsymbol{x})$, on the other hand, is an probabilistic function in continuous form. The mathematical expression of $\pi(\boldsymbol{x})$ is written as:

$$\pi(\boldsymbol{x}) = \Phi\left(\frac{\mu(\boldsymbol{x}) - T_{spec}}{\sigma(\boldsymbol{x})}\right) \tag{5.7}$$

where $\Phi(\cdot)$ is the cumulative distribution function of standard normal distribution and T_{spec} represents the threshold of design specification.

Next, we introduce quasi-optimal sampling distribution $g(\boldsymbol{x})$ defined as:

$$g(\boldsymbol{x}) = \frac{\pi(\boldsymbol{x})f(\boldsymbol{x})}{\int \pi(\boldsymbol{x})f(\boldsymbol{x})d\boldsymbol{x}} \tag{5.8}$$

We notice that Equation (5.8) has the exact same format as $g^{opt}(\boldsymbol{x})$, but in a probabilistic perspective. For a particular GP meta-model, sampling from corresponding quasi-optimal distribution has the highest possibility to obtain "important" samples, which locate on failure region boundaries.

5.3 Adaptive Sampling Strategy

According to Equation (5.8), the effectiveness of sampling distribution is strongly dependent on the accuracy of our GP meta-model. However, it is challenging to directly train such GP that can characterize indicator function $I(\boldsymbol{x})$. And the computational cost explodes as circuit complexity increases. To address this issue, we propose to improve our model accuracy through a stepwise adaptive sampling strategy.

Our sampling scheme starts with LHS method. It is extensively applied as a space-filling sampling method, which samples from evenly partitioned multidimensional parametric space.

Next, our MIS adaptive sampling strategy is depicted in Figure 5.1. It consists of two major steps. In the sample propagation step, at certain iteration T, we first construct quasi-optimal intermediate sampling distribution $g_T(\boldsymbol{x})$ based on our GP model. Then we implement MCMC sampling to choose the next most informative sample point $\boldsymbol{x}_T$. The Metropolis-Hasting sampler we applied here is detailed in Algorithm 1. Note that generation and acceptance of samples are independently performed, which enables us to obtain vectorized result. In the following model calibration step, we enrich our training sample set with $\boldsymbol{x}_T$, and calibrate our meta-model, as shown in Figure 5.1(b). Our adaptive sampling procedure iterates between these two steps until the failure probability estimation converge to a stable value with specific confidence interval. In this way, our GP meta-model is refined immediately after sample propagation, providing highest flexibility to explore rare failure regions with limited budget.

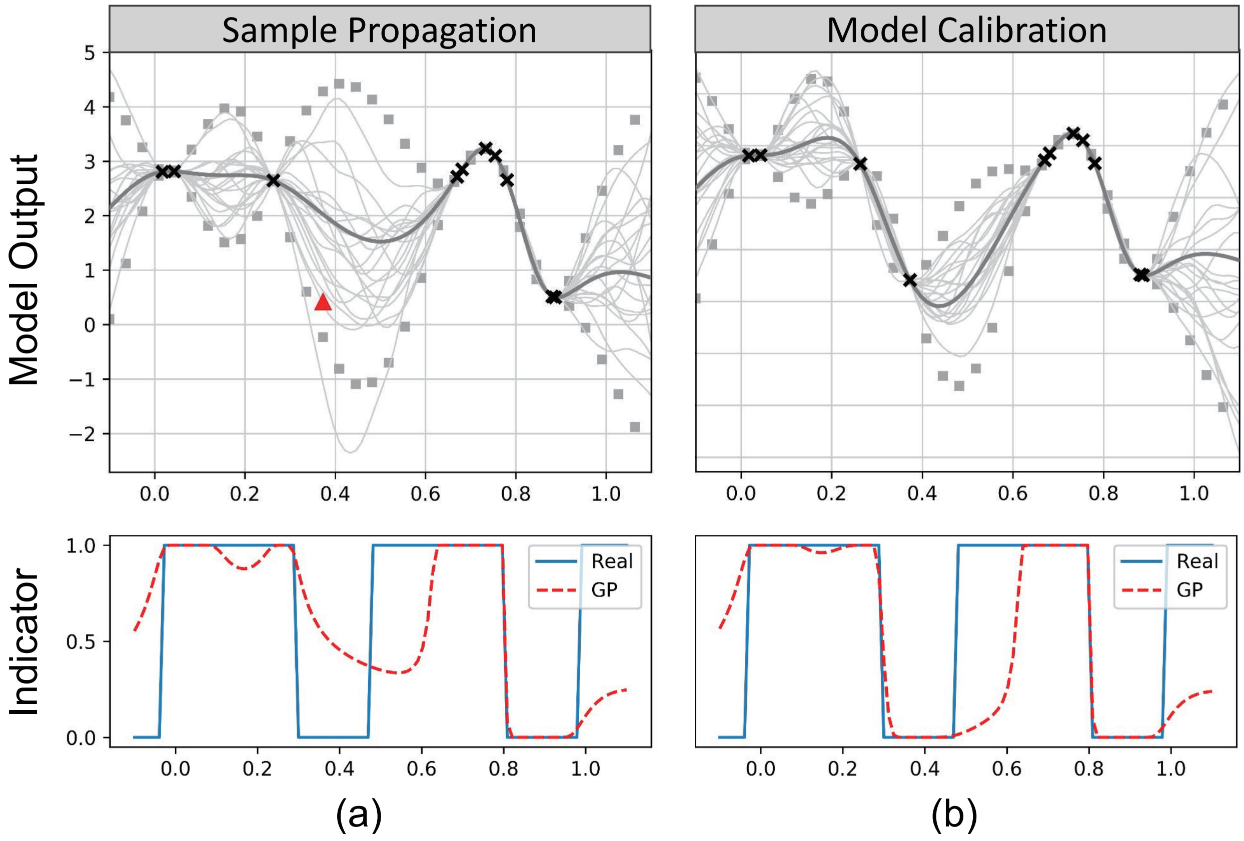


Figure 5.1: The calibration of GP model through adaptive sampling strategy. The dark crosses are training sample set, and red triangle represents the next sample from MCMC simulation. The dark gray lines show the mean value of GP models, and light gray lines indicate posterior distributions of trained GP model. Dashed lines mark the 95% confidence interval of GP model, which is proportional to prediction variance.

5.4 MIS Estimator Analysis

The major contribution of our MIS algorithm is that we develop a novel estimator. It is generated by updating a series of intermediate estimators, which can successively provide estimation from a cloud of iterative sampling distributions. Compared with sampling from a single IS distribution, we can eliminate the time-consuming static IS procedure, and can accurately locate failure region boundaries.

Based on our sampling strategy described in Section III, until certain iteration T, sample set $\{\boldsymbol{x}_t\}_{t=1}^{T}$ has been generated from all the previous sampling distributions $\{g_t(\boldsymbol{x})\}_{t=1}^{T}$. For

Algorithm 7: Metropolis-Hastings sampler

```
Input: Initial sample $\boldsymbol{x}^{(0)}$, proposal normal PDF $p(\boldsymbol{x})$,
        length of Markov chain $M$, target PDF $g_T(\boldsymbol{x})$
Output: $\boldsymbol{x}_T$
$i = 0$
while $i \leq M$ do
    generate a new sample $\boldsymbol{x}^* \sim p(\cdot|\boldsymbol{x}^{(i)})$
    calculate acceptance rate
        $$\alpha^{(i+1)} = \min\left(1; \frac{g_T(\boldsymbol{x}^*)p(\boldsymbol{x}^{(i)}|\boldsymbol{x}^*)}{g_T(\boldsymbol{x}^{(i)})p(\boldsymbol{x}^*|\boldsymbol{x}^{(i)})}\right)$$
    generate $u \sim U[0,1]$
    if $u \leq \alpha^{(i+1)}$ then
        accept new sample $\boldsymbol{x}^{(i+1)} = \boldsymbol{x}^*$
    else
        reject new sample $\boldsymbol{x}^{(i+1)} = \boldsymbol{x}^{(i)}$
    end
end
return $\boldsymbol{x}_T \sim g_T(\boldsymbol{x})$
```

current sample point $\boldsymbol{x}_T$, we perform Deterministic Mixture (DM) strategy to construct its intermediate estimator. We start by constructing the mixture density by averaging previous sampling distributions $\frac{1}{T}\sum_{t=1}^{T} g_t(\boldsymbol{x}_T)$. Thereby, its importance weight $w(\boldsymbol{x_T})$ is defined as the ratio between original probability density $f(\boldsymbol{x_T})$ and mixture density:

$$w(\boldsymbol{x_T}) = \frac{f(\boldsymbol{x_T})}{\frac{1}{T}\sum_{t=1}^{T} g_t(\boldsymbol{x}_T)} \tag{5.9}$$

This importance weight can actually be regarded as the extension of conventional static IS importance weight defined in Equation (2.5). Thus the intermediate estimator $\hat{P}_{fail,T}^{MIS}$ for iteration T is written as:

$$\hat{P}_{fail,T}^{MIS} = \frac{1}{T}\sum_{t=1}^{T} w(\boldsymbol{x}_t)I(\boldsymbol{x}_t) \tag{5.10}$$

As iteration continues, this intermediate estimator keeps updating until the termination of sampling procedure, giving final global estimation on failure probability.

5.4.1 Adaptation of MIS Estimator

The adaptive mechanism of MIS is driven by the uncertainty from the intermediate estimators. To be specific, each sample $\boldsymbol{x}_t$ comes from a corresponding distribution $g_t(\boldsymbol{x})$ that can describe local features of the original distribution $f(\boldsymbol{x})$. After we perform SPICE simulation on typical sample point and use it to calibrate our meta-model, a random walk will be generated toward regions of higher probabilities. That is, regions with larger mismatches or regions with higher importance. As our meta-model is becoming more and more accurate, our sampling distribution heuristically approaches failure boundaries, where the importance weight $w(\boldsymbol{x_T})$ is largest. Thus more observations are added to the sample set, and our final sampling distribution is increasingly similar to the ideal failure event distribution $g^{opt}(\boldsymbol{x})$.

Another good property of our MIS estimator is that the adaptation procedure is independent of multiple families of sampling distributions. It means that MIS has the potential to allow multiple meta-models adapting in parallel, in order to explore different regions. Therefore, our MIS method can tackle circuit cases with complex failure regions, or with various failure mechanisms.

5.4.2 Unbiasedness and Variance of MIS Estimator

In this section, we first prove that our MIS estimator $\hat{P}_{fail}^{MIS}$ is unbiased as iteration proceeds. The unbiasedness can be validated by its expected value, which guarantees that $\hat{P}_{fail}^{MIS}$ converges to the ground truth failure probability:

$$\begin{aligned} E\left[\hat{P}_{fail}^{MIS}\right] &= E\left[\frac{1}{T}\sum_{t=1}^{T} w(\boldsymbol{x}_t)I(\boldsymbol{x}_t)\right] \\ &= \frac{1}{T}\sum_{t=1}^{T} E\left[\frac{f(x)I(x)}{\frac{1}{T}\sum_{t=1}^{T} g_t(x)}\right] \\ &= \frac{1}{T}\sum_{t=1}^{T}\int \frac{f(x)I(x)}{\frac{1}{T}\sum_{j=1}^{T} g_j(x)} g_t(x)dx = P_{fail} \end{aligned} \tag{5.11}$$

Next, we demonstrate that proposed MIS estimator is strictly superior or equal to static IS estimator in terms of lower variance. The worst-case scenario of MIS degenerates into static IS, which occurs at iteration $T = 1$. Mathematically, the variance of two estimators is given by:

$$Var(\hat{P}_{fail}^{MIS}) = \frac{1}{T^2}\sum_{i=1}^{T}\left(\int \frac{f^2(\boldsymbol{x})I^2(\boldsymbol{x})}{\frac{1}{T}\sum_{j=1}^{T} g_j(\boldsymbol{x})} - P_{fail}^2\right) \tag{5.12}$$

$$Var(\hat{P}_{fail}^{IS}) = \frac{1}{T^2}\sum_{i=1}^{T}\left(\int \frac{f^2(\boldsymbol{x})I^2(\boldsymbol{x})}{g_i(\boldsymbol{x})} - P_{fail}^2\right) \tag{5.13}$$

By subtracting Equation (5.12) and (5.13), we prove that $Var(\hat{P}_{fail}^{MIS})$ is always smaller by deriving the following inequality:

$$\begin{aligned} \frac{1}{\frac{1}{T}\sum_{j=1}^{T} g_j(\boldsymbol{x})} &= \frac{1}{\frac{T-1}{T}\frac{1}{T-1}\sum_{i=1}^{T-1} g_i(\boldsymbol{x}) + \frac{1}{T}g_T(\boldsymbol{x})} \\ &\leq \frac{(T-1)/T}{\frac{1}{T-1}\sum_{i=1}^{T-1} g_i(\boldsymbol{x})} + \frac{1/T}{g_T(\boldsymbol{x})} \\ &\leq \frac{T-1}{T}\frac{1}{T-1}\sum_{i=1}^{T-1}\frac{1}{g_i(\boldsymbol{x})} + \frac{1}{T}\frac{1}{g_T(\boldsymbol{x})} \\ &= \frac{1}{T}\sum_{i=1}^{T}\frac{1}{g_i(\boldsymbol{x})} \end{aligned} \tag{5.14}$$

With the unbiased estimation and lower variance, our MIS method shall exhibit better stability and convergence. It is validated by the accuracy and efficiency comparison in the experiments of Section IV.

5.5 Experiment Results

In this section, our proposed yield analysis algorithm is first tested on a typical 6T SRAM bit cell with 36 variables. For analog yield analysis, we then validate our MIS on a two-stage operational transimpedance amplifier (OTA) with 84 variables. We implement MC as ground truth for accuracy comparison. To show the efficiency of MIS, we also implement several state-of-the-art approaches including Hyperspherical Clustering and Sampling (HSCS) [WBH16] and Adaptive Importance Sampling (AIS) [SLYH18]. The SPICE model is SMIC 40nm transistor model. All the experiments are performed on Linux server with Intel Xeon X5675 CPU @3.07 GHz and 94 GB RAM.

5.5.1 Experiments on 6T SRAM bit bell

The schematic of typical 6T SRAM bit cell is shown in Figure 5.2. Four transistors MP1, MN2, MP3 and MN4 form two cross-coupled inverters and use two steady states (either '0' or '1') to store data in this cell. The other two transistors MN5 and MN6 work as switches to control access to the storage cell during reading, writing and standby operations. Taking reading failure as an example, it occurs when the voltage difference between BL and BLB is too small to be captured by sense amplifier in a certain period. The performance of the circuit is characterized by the delay of discharging bitline, which should be smaller than a given threshold for reading success. We implement different methods to compare their accuracy and efficiency.

5.5.1.1 Accuracy Comparison

To evaluate the accuracy of difference methods, we introduce Figure of Merit (FOM), ρ, to characterize the convergence and confident interval of our estimation. Its definition is:

$$\rho = \frac{\sqrt{\sigma^2_{\hat{p}_{fail}}}}{\hat{p}_{fail}} \tag{5.15}$$

where $\hat{p}_{fail}$ represents the failure probability and $\sigma_{\hat{p}_{fail}}$ denotes its standard deviation. To clarify, if an estimator terminates with $\rho \leq \epsilon\sqrt{log(1/\delta)}$, we can claim that $\hat{p}_{fail}$ is $(1-\epsilon) \times$

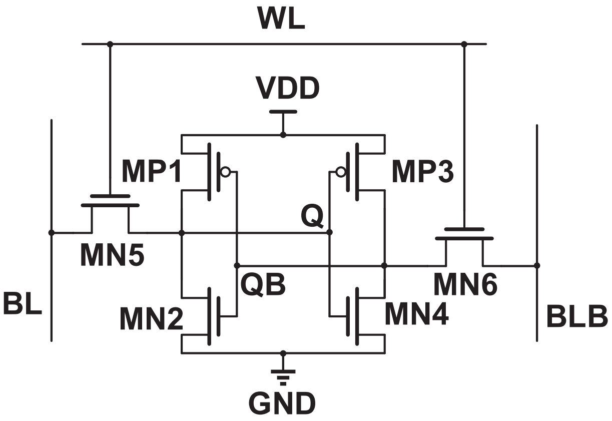


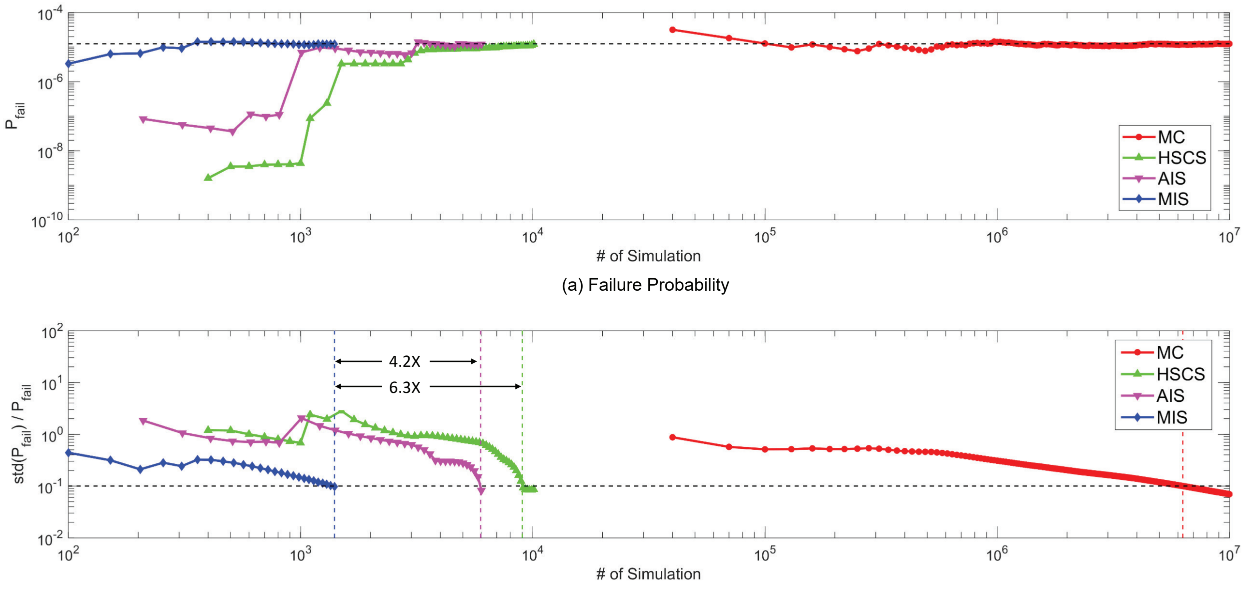
Figure 5.2: The schematic of typical 6T SRAM bit cell

100% accurate with $(1-\delta) \times 100\%$ confidence. To guarantee the accuracy of estimation, we set $\rho = 0.1$ as the convergence criterion to reveal that the estimation reaches 90% accuracy level with 90% confidence interval. It is depicted as dashed line in Figure 5.3. We observe that the estimation of HSCS, AIS and proposed MIS all succeed to match the ground truth value when sufficient simulations are available.

More detailed comparison of those approaches is illustrated in Table 5.1. We note that the proposed MIS algorithm provides the most accurate estimation with only 3.2% relative error, while HSCS and AIS have 4.6% and 6.1% relative error, respectively. It is because MIS iteratively improves the optimal sampling distribution by searching for failure region, and accurately determine the failure boundaries through Gaussian Process model to avoid misclassified samples.

Table 5.1: Accuracy and efficiency comparison on 36 dimensional SRAM bit cell

	MC	HSCS	AIS	MIS
Failure prob.	1.24e-5	1.18e-5	1.32e-5	1.20e-5
Relative error	golden	4.6%	6.1%	3.2%
# Sim. runs	6.3e6	9019	5962	1413
Speedup	1X	698X	1056X	4458X



(a) Failure Probability

(b) Figure of Merit

Figure 5.3: Evolution comparison of failure prob. and FOM on SRAM bit cell

5.5.1.2 Efficiency Comparison

Figure 5.3 shows the efficiency of MC, HSCS, AIS and MIS. The proposed MIS algorithm can provide fastest convergence. It is attributed to the adaptive sampling strategy applied in MIS, which is far more efficient than static sampling method in HSCS and resampling scheme in AIS. As detailed in Table 5.1, HSCS and AIS require 9019 and 5962 samples converge to golden reference, respectively. With 4458X, 6.4X and 4.2X speedup over MC, HSCS and AIS method, MIS obtains a higher accuracy with only 1413 simulations.

5.5.2 Experiments on Two-Stage Amplifier

In this section, we verify that the proposed method is capable of handling problems on an analog circuit with multiple performance metrics. Figure 5.4 shows the circuit schematic of two-stage OTA using master-slave structure for low supply voltage application. The slave stage consists of the tail current transistor (*MP5*), the differential pair (*MP1* and *MP2*) and the current-mirror load (*MN1* and *MN2*). The master stage replicates the tail current source and the transconductance transistor of the slave stage circuit (i.e. *MP3*, *MP4*, *MP6* and *MN3* are the copies of *MP1*, *MP2*, *MP5* and *MN1*, respectively). *MP5* and *MP6* operate in linear region to save voltage margin. A robust OTA design should satisfy multiple specification requirements. In our experiments setting, we consider various performance specifications, including voltage gain margin, gain bandwidth, phase margin and 3dB bandwidth. There are in total 84 variation parameters in this case.

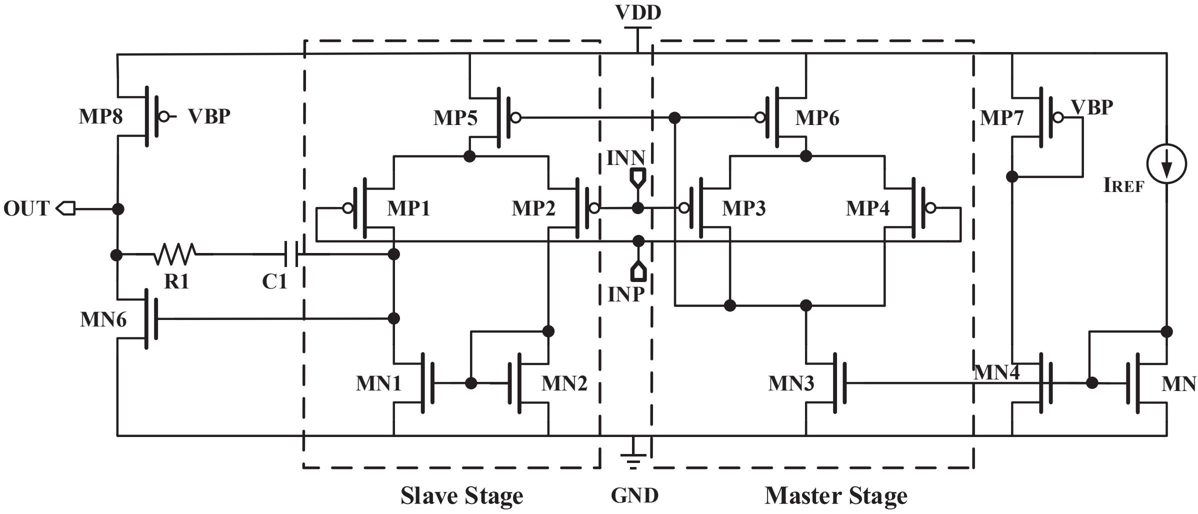
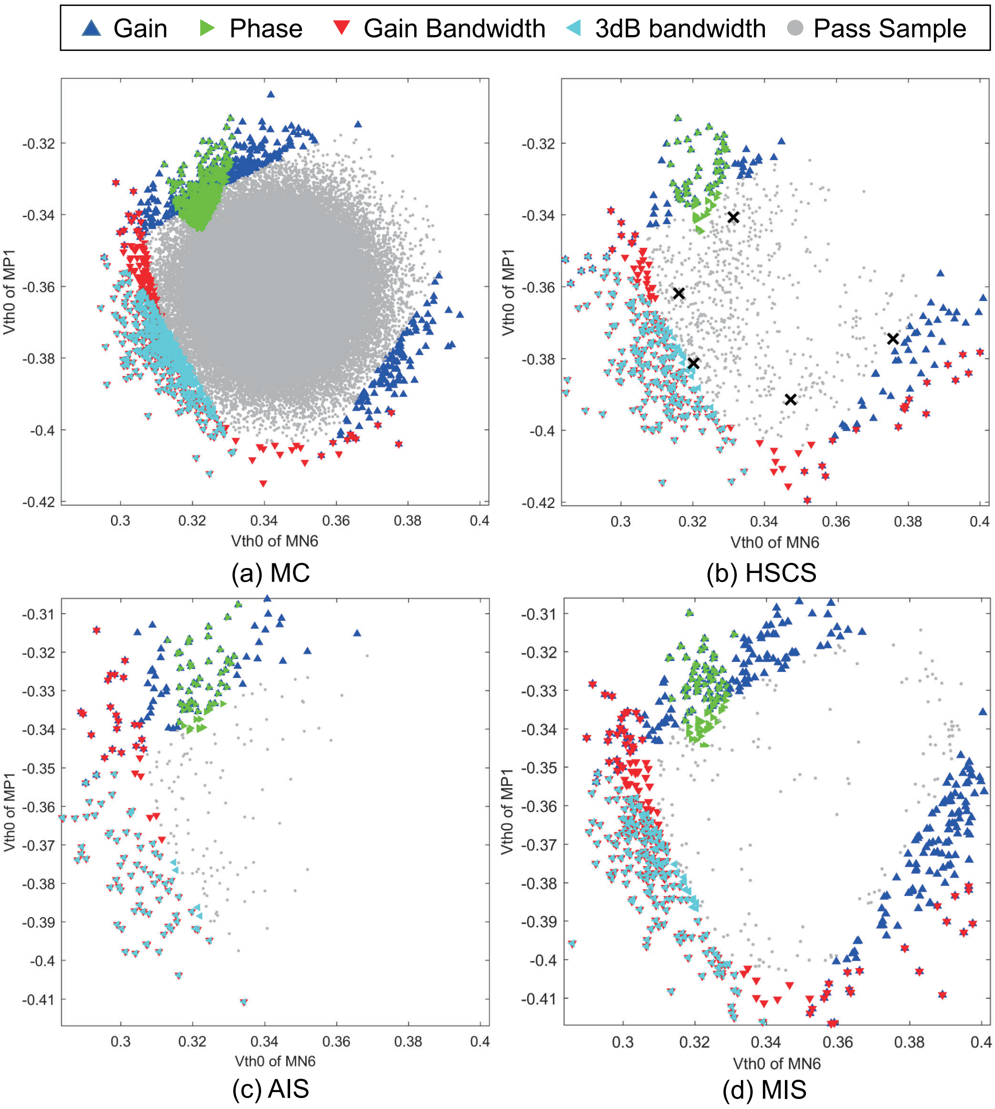


Figure 5.4: The schematic of two-stage operational transimpedance amplifier

5.5.2.1 Comparison of Visualized Failure Regions

In order to compare the capability of locating multiple failure regions, we project the sample points onto two most important dimensions, which are the threshold voltage of MN6 and MP1. Figure 5.5 shows the visualized sample points in 2D parametric space for different methods. Here gray dots denote passed samples and multi-colored triangles are samples with different failure schemes. Based on ground truth MC sampling result shown in Figure 5.5(a), we notice that the failure regions are overlapped and have complex non-convex boundaries.

In Figure 5.5(b), HSCS method performs K-means algorithm to group samples into clusters, and generates a set of min-norm points as centroids for each cluster. Then it performs multiple spherical sampling to shift sample mean to these min-norm points. Although it is able to cover the majority of failure regions, the spherical sampling strategy lacks flexibility



(a) MC

(b) HSCS

(c) AIS

(d) MIS

Figure 5.5: Multiple failure regions coverage comparison. (samples with different failure schemes are colored triangles, dark crosses in HSCS denote the shifted mean)

to characterize the boundaries. Also, we observe that large proportion of samples fall in the passed regions, which makes it inefficient.

Figure 5.5(c) displays the failure samples collected with AIS method. We notice that AIS cannot detect all the failure regions, which gives relatively smaller failure probability. This property is caused by weight degeneracy during resampling procedure. As iteration proceeds, samples with smaller weights are more likely to be neglected, which leads to missing failure regions on the right hand side.

As shown in Figure 5.5(d), our MIS method considers each performance metric separately. We first build four initial GP meta-models between each performance metric and input variation vector in a parallel fashion. Next, for each meta-model, we approximate its failure event indicator function $\{I_i^{(0)}(\boldsymbol{x})\}_{i=1}^4$ and generate intermediate sampling distribution $\{g_i^{(0)}(\boldsymbol{x})\}_{i=1}^4$. At each iteration t, four samples are collected from each intermediate distribution, formulating a mixture sampling distribution $G^{(t)}(\boldsymbol{x})$ by averaging out $\{g_i^{(t)}(\boldsymbol{x})\}_{i=1}^4$. This concept is actually very similar to the methodology in Kernel Density Estimation (KDE) [Sil18]. As displayed in Figure 5.5(d), the whole sample set of MIS is generated by incrementally sampling from all the previous intermediate sampling distribution $g_i^{(t)}(\boldsymbol{x})$. We notice that the MIS sample set is able to capture all the boundaries for different failure schemes, preventing biased estimation on P_{fail} as seen in AIS.

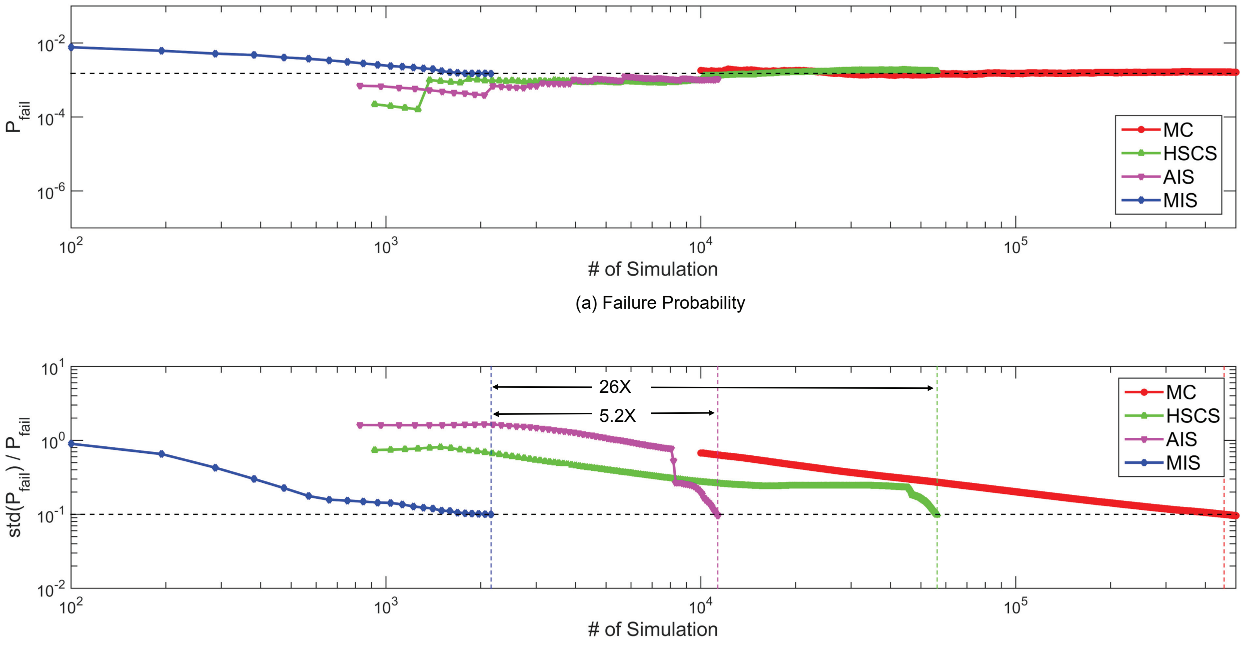
Another observation from Figure 5.5 is that, for this amplifier circuit with multiple failure schemes, our MIS is prone to sample from the distinct boundaries for different performance metrics. That is, our formulated mixture sampling distribution has slight density deviation compared with the optimal sampling distribution. The optimal distribution, which is ideal failure event density, is shown as colored triangles in MC. However, instead of training complex global meta-model, our experiment result verifies that training separate meta-model can speed up with order of magnitude. We notice that among all these methods, MIS requires the smallest number of sample points to guarantee all failure-region coverage. Proposed GP meta-model is extremely simple to train with minimum sacrifice in sampling efficiency.

5.5.2.2 Accuracy and Efficiency Comparison

We evaluate proposed MIS and other methods (MC, HSCS, AIS) and compare their efficiencies and accuracies on two-stage amplifier to validate the improvement of proposed method.

Table 5.2 shows the analysis performance comparison of Monte Carlo, HSCS, AIS and proposed MIS. According to results of Monte Carlo, the standard failure probability of two-stage amplifier is higher than SRAM bit cell case since its multi-performance requirements and more complex failure regions. Under this situation, the yield analysis accuracy and efficiency of both HSCS and AIS are diminished with over 10% relative error and under 100X speedup rate, while proposed method remains relatively stable performance of 8% relative error and 589X speedup.

The comparison validates that MIS can be adapted to wide-range failure probability and multiple performances yield analysis. This extendability is contributed by building separate meta-models for each specification to locate sampling areas instead of brute fail/success classification, which will be further evaluated and analyzed in next section.



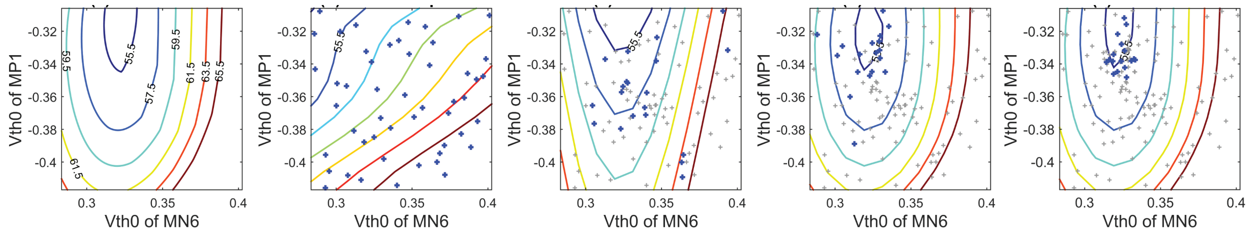
(a) Failure Probability

(b) Figure of Merit

Figure 5.6: Evolution comparison of failure prob. and FOM on Two-stage OTA

Table 5.2: Accuracy and efficiency comparison on 84 dimensional Two-Stage OTA

	MC	HSCS	AIS	MIS
Failure prob.	1.5e-3	1.68e-3	1.05e-3	1.60e-3
Relative error	golden	12%	30%	6.7%
# Sim. runs	458700	56290	11316	2156
Speedup	1X	8X	41X	213X



(a) Monte Carlo (b) Initial sample set → (c) Iteration 20 → (d) Iteration 40 → (e) Iteration 60

Figure 5.7: 2D visualized plot of proposed adaptive sampling strategy. Here contour lines denote boundaries of different OTA phase margin threshold. Blue crosses are current sample set to calibrate contour lines, while gray crosses stand for all previous samples.

5.5.2.3 MIS Adaptation for Poor Initialization

In this section, we demonstrate that the adaption of MIS enables it to recover from poor initialization states. Here we take the amplifier phase margin as target metric. Figure 5.7(a) displays the ground truth MC contour plot, where colored contour lines represent different threshold for phase margin. Among these contour lines, the black one with 55.5 marked on it is the failure region. In Figure 5.7(b)-(e), we show the evolution of contour lines generated from intermediate GP meta-model. The incremental sample set used to calibrate model is also depicted. To be specific, Figure 5.7(b) shows a poor initial sample set generated from space-filling LHS method. As iteration proceeds, most samples, through a random walk, have the trend to approach the failure region boundary. Other few samples are generated near the regions where prediction mismatch is largest. After 60 iterations, our meta-model prediction is very similar compared with golden MC, which implies that it can output accurate indicator function $I(\boldsymbol{x})$. Our sampling distribution is thus stabilized around failure region boundaries, providing constant and robust estimation.

5.6 Conclusions

In this chapter, we propose a meta-model based importance sampling (MIS) to tackle the challenging circuit reliability problems with multiple disjoint failure regions. We first apply Gaussian Process meta-model to construct quasi-optimal sampling distribution. Next, we design a novel adaptive sampling strategy to generate new samples from a set of intermediate distributions, and update the failure probability at the end of each iteration. The experimental results demonstrate that the proposed MIS algorithm can provide extremely high accuracy and efficiency. For SRAM bit cell with 36 variables, MIS achieves 4458X speedup over MC and 4-6X over other state-of-the-art methods. For 84-dimensional two-stage OTA with multiple failure schemes, MIS is 213X faster than MC, while other approaches fail to provide a reasonable accuracy.

CHAPTER 6

Summary

As the microelectronic devices scale down to nanometer, various memory and analog circuits are more vulnerable to process, voltage and temperature (PVT) variations. Therefore, circuit parameters can deviate significantly from their nominal values specified by designers, which make the circuit performance merits may fail the design specifications under the nominal condition. The scale of circuit yield analysis problem is a high-sigma problem, which cannot be solved by traditional deterministic approaches. Instead, people turn to statistical circuit simulation method. It helps circuit designers to debug circuits in the pre-silicon phase considering PVT variations, which can dramatically shorten the required time to market. This dissertation thesis present several pieces of research works related to statistical yield analysis.

In Chapter 2, we propose an adaptive importance sampling (AIS) algorithm. AIS has several iterations of sampling region adjustments, while existing methods pre-decide a static sampling distribution. We design two adaptive frameworks based on Resampling and population Metropolis-Hastings (MH) to iteratively search for failure regions. The experimental results of AIS method exhibit better efficiency and higher accuracy. For SRAM bit cell with single failure region, AIS method uses 2-27× fewer samples and reaches better accuracy when compared to several recent methods. For a two-stage amplifier circuit with multiple failure regions, AIS method is 90× faster than Monte Carlo and 7-23.× over other methods. For charge pump circuit and C^2MOS master-slave latch circuit, AIS method can reach 6-18× and 4-6× speedup over other methods, respectively.

In Chapter 3, we develop an Adaptive Clustering and Sampling (ACS) method. ACS proceeds iteratively to cluster samples and adjust sampling distribution, while most existing approaches pre-decide a static sampling distribution. By adaptively searching in multiple cone-shaped subspaces, ACS obtains better accuracy and efficiency. This result is validated by our experiments. For SRAM bit cell with single failure region, ACS requires 3-5X fewer samples and achieves better accuracy compared with existing approaches. For 576-dimensional SRAM column circuit with multiple failure regions, ACS is 2050X faster than MC without compromising accuracy, while other methods fail to converge to correct failure probability in our experiment.

In Chapter 4, we develop a meta-model using Low-Rank Tensor Approximation (LRTA) to substitute expensive SPICE simulation. The polynomial degree of our LRTA model grows linearly with circuit dimension. This makes it especially promising for high-dimensional circuit problems. Our LRTA meta-model is solved efficiently with a robust greedy algorithm, and calibrated iteratively with an adaptive sampling method. Experiments on bit cell and SRAM column validate that proposed LRTA method outperforms other state-of-the-art approaches in terms of accuracy and efficiency.

In Chapter 5, we develop a novel meta-model based importance sampling (MIS) method. MIS utilizes Gaussian Process meta-model to construct quasi-optimal importance sampling distribution, and performs Markov Chain Monte Carlo (MCMC) simulation to generate new samples from the proposed distribution. By updating our global Importance Sampling estimator in an iterated framework, MIS leads to better efficiency and higher accuracy. For SRAM bit cell with single failure region, MIS uses 4-6X fewer samples and reaches better accuracy when compared to several recent methods. For a two-stage amplifier circuit with multiple failure schemes, MIS is 213X faster than MC without compromising accuracy, while other methods fail to cover all failure regions in our experiment.

REFERENCES

[Bro98] Stephen Brooks. Markov chain monte carlo method and its application. *Journal of the royal statistical society: series D (the Statistician)*, 47(1):69–100, 1998.

[CCS04] Runzi Chang, Yu Cao, and Costas J Spanos. Modeling the electrical effects of metal dishing due to cmp for on-chip interconnect optimization. *IEEE transactions on electron devices*, 51(10):1577–1583, 2004.

[CGMR04] Olivier Cappé, Arnaud Guillin, Jean-Michel Marin, and Christian P Robert. Population monte carlo. *Journal of Computational and Graphical Statistics*, 13(4):907–929, 2004.

[Che99] Song Xi Chen. Beta kernel estimators for density functions. *Computational Statistics & Data Analysis*, 31(2):131–145, 1999.

[DCB$^+$10] Guillaume Desjardins, Aaron Courville, Yoshua Bengio, Pascal Vincent, and Olivier Delalleau. Tempered markov chain monte carlo for training of restricted boltzmann machines. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 145–152, 2010.

[DM03] Patrick G Drennan and Colin C McAndrew. Understanding mosfet mismatch for analog design. *IEEE Journal of solid-state circuits*, 38(3):450–456, 2003.

[DQSC08] Lara Dolecek, Masood Qazi, Devavrat Shah, and Anantha Chandrakasan. Breaking the simulation barrier: Sram evaluation through norm minimization. In *2008 IEEE/ACM International Conference on Computer-Aided Design*, pages 322–329. IEEE, 2008.

[EBSLM97] Martin Eisele, Jörg Berthold, Doris Schmitt-Landsiedel, and Reinhard Mahnkopf. The impact of intra-die device parameter variations on path delays and on the design for yield of low voltage digital circuits. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 5(4):360–368, 1997.

[EHJ$^+$04] Bradley Efron, Trevor Hastie, Iain Johnstone, Robert Tibshirani, et al. Least angle regression. *The Annals of statistics*, 32(2):407–499, 2004.

[EMLB17] Víctor Elvira, Luca Martino, David Luengo, and Mónica F Bugallo. Improving population monte carlo: Alternative weighting and resampling schemes. *Signal Processing*, 131:77–91, 2017.

[GP08] Puneet Gupta and Evanthia Papadopoulou. Yield analysis and optimization., 2008.

[GYH11] Fang Gong, Hao Yu, and Lei He. Stochastic analog circuit behavior modeling by point estimation method. In *Proceedings of the 2011 international symposium on Physical design*, pages 175–182, 2011.

[KJN06] Rouwaida Kanj, Rajiv Joshi, and Sani Nassif. Mixture importance sampling and its application to the analysis of sram designs in the presence of rare failure events. In *2006 43rd ACM/IEEE Design Automation Conference*, pages 69–72. IEEE, 2006.

[KLW94] Augustine Kong, Jun S Liu, and Wing Hung Wong. Sequential imputations and bayesian missing data problems. *Journal of the American statistical association*, 89(425):278–288, 1994.

[LBD15] Tiancheng Li, Miodrag Bolic, and Petar M Djuric. Resampling methods for particle filtering: classification, implementation, and strategies. *IEEE Signal processing magazine*, 32(3):70–86, 2015.

[Li10] Xin Li. Finding deterministic solution from underdetermined equation: large-scale performance variability modeling of analog/rf circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 29(11):1661–1668, 2010.

[LLGP04] Xin Li, Jiayong Le, Padmini Gopalakrishnan, and Lawrence T Pileggi. Asymptotic probability extraction for non-normal distributions of circuit performance. In *IEEE/ACM International Conference on Computer Aided Design, 2004. ICCAD-2004.*, pages 2–9. IEEE, 2004.

[Moo19] Samuel K Moore. Another step toward the end of moore's law: Samsung and tsmc move to 5-nanometer manufacturing-[news]. *IEEE Spectrum*, 56(6):9–10, 2019.

[Nas01] Sani R Nassif. Modeling and analysis of manufacturing variations. In *Proceedings of the IEEE 2001 Custom Integrated Circuits Conference (Cat. No. 01CH37169)*, pages 223–228. IEEE, 2001.

[QTD+10] Masood Qazi, Mehul Tikekar, Lara Dolecek, Devavrat Shah, and Anantha Chandrakasan. Loop flattening & spherical sampling: Highly efficient model reduction techniques for sram yield analysis. In *2010 Design, Automation & Test in Europe Conference & Exhibition (DATE 2010)*, pages 801–806. IEEE, 2010.

[Sil18] Bernard W Silverman. *Density estimation for statistics and data analysis.* Routledge, 2018.

[SLYH18] Xiao Shi, Fengyuan Liu, Jun Yang, and Lei He. A fast and robust failure analysis of memory circuits using adaptive importance sampling method. In *2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2018.

[SR07] Amith Singhee and Rob A Rutenbar. Statistical blockade: a novel method for very fast monte carlo simulation of rare circuit events, and its application. In *2007 Design, Automation & Test in Europe Conference & Exhibition*, pages 1–6. IEEE, 2007.

[SR09] Amith Singhee and Rob A Rutenbar. Statistical blockade: Very fast statistical simulation and modeling of rare circuit events and its application to memory design. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 28(8):1176–1189, 2009.

[Sud08] Bruno Sudret. Global sensitivity analysis using polynomial chaos expansions. *Reliability engineering & system safety*, 93(7):964–979, 2008.

[SWCR08] Amith Singhee, Jiajing Wang, Benton H Calhoun, and Rob A Rutenbar. Recursive statistical blockade: An enhanced technique for rare event simulation with application to sram circuit design. In *21st International Conference on VLSI Design (VLSID 2008)*, pages 131–136. IEEE, 2008.

[SWMW89] Jerome Sacks, William J Welch, Toby J Mitchell, and Henry P Wynn. Design and analysis of computer experiments. *Statistical science*, pages 409–423, 1989.

[SYH$^+$19] Xiao Shi, Hao Yan, Qiancun Huang, Jiajia Zhang, Longxing Shi, and Lei He. Meta-model based high-dimensional yield analysis using low-rank tensor approximation. In *Proceedings of the 56th Annual Design Automation Conference 2019*, pages 1–6, 2019.

[Tib96] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[VWG06] Sarma Vrudhula, Janet Meiling Wang, and Praveen Ghanta. Hermite polynomial based interconnect analysis in the presence of process variations. *IEEE Transactions on Computer-Aided Design of Integrated circuits and systems*, 25(10):2001–2011, 2006.

[WBH16] Wei Wu, Srinivas Bodapati, and Lei He. Hyperspherical clustering and sampling for rare event analysis with multiple failure region coverage. In *Proceedings of the 2016 on International Symposium on Physical Design*, pages 153–160, 2016.

[WGCH14] Wei Wu, Fang Gong, Gengsheng Chen, and Lei He. A fast and provably bounded failure analysis of memory circuits in high dimensions. In *2014 19th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 424–429. IEEE, 2014.

[WLY$^+$18] Mengshuo Wang, Wenlong Lv, Fan Yang, Changhao Yan, Wei Cai, Dian Zhou, and Xuan Zeng. Efficient yield optimization for analog and sram circuits via gaussian process regression and adaptive yield estimation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(10):1929–1942, 2018.

[WXK$^+$14] Wei Wu, Wenyao Xu, Rahul Krishnan, Yen-Lung Chen, and Lei He. Rescope: High-dimensional statistical circuit simulation towards full failure region coverage. In *Proceedings of the 51st Annual Design Automation Conference*, pages 1–6, 2014.

[WYL$^+$16] Mengshuo Wang, Changhao Yan, Xin Li, Dian Zhou, and Xuan Zeng. High-dimensional and multiple-failure-region importance sampling for sram yield analysis. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 25(3):806–819, 2016.

[XK02] Dongbin Xiu and George Em Karniadakis. The wiener–askey polynomial chaos for stochastic differential equations. *SIAM journal on scientific computing*, 24(2):619–644, 2002.

[YYW14] Jian Yao, Zuochang Ye, and Yan Wang. An efficient sram yield analysis and optimization method with adaptive online surrogate modeling. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 23(7):1245–1253, 2014.