An Efficient Bayesian Yield Estimation Method for High Dimensional and High Sigma SRAM Circuits

Jinyuan Zhai[1], Changhao Yan[*1], Sheng-Guo Wang[2], Dian Zhou[*1,3]
[1] ASIC & System State Key Lab, Dept. of Microelectronics, Fudan University, Shanghai, China,
[2] Dept. of ECE, University of North Carolina at Charlotte, Charlotte, USA,
[3] Dept. of EE, University of Texas at Dallas, Dallas, USA

ABSTRACT

With increasing dimension of variation space and computational intensive circuit simulation, accurate and fast yield estimation of realistic SRAM chip remains a significant and complicated challenge. In this paper, du Experiment results show that the proposed method has an almost constant time complexity as the dimension increases, and gains 6x speedup over the state-of-the-art method in the 485D cases.

1. INTRODUCTION

As semiconductor fabrication technology shrinks to nanometer scale, process variances, such as doping fluctuation and threshold voltage variation, have significant impacts on performance and reliability of circuits. Meanwhile, the SRAM bit-cell is generally designed in the minimum size of technology nodes, which makes it vulnerable to process variations.

As realistic SRAM bit-cells are replicated into a large array, for a memory chip with moderate yield, the failure rate of an SRAM bit-cell is required to be extremely low (smaller than 10^{-6}). Therefore, the main challenge is the high sigma property [7] of yield of SRAM circuits. Conventional Monte Carlo (MC) method becomes extremely inefficient for high sigma SRAM yield analysis.

One challenge of the yield estimation of SRAM circuits is the extreme high dimensions, which is recently addressed by [4], [10]-[12]. Another severe and indispensable challenge is the simulation time of SPICE. For an SRAM column with 80 SRAM bit-cells, the CPU time of a transient simulation by SPICE is about 2 hours in our experiment. Actually, transient simulation time of SPICE is roughly $O(n^3)$ for SRAM circuits, where n is the number of bit-cells. Therefore, SPICE simulations become extremely time-consuming in high dimensional cases. Though the modeling technical of SRAM bit-cells can sharply decrease the simulation time, it is hardly acceptable by industry and its validity needs further verifications. Therefore, the key problem is how to decrease the number of SPICE simulations of high dimensional circuits.

To address the efficiency issue of MC, efficient and accurate methods are imperative to estimate SRAM failure rates. Generally, these methods can be divided into two categories: Importance Sampling (IS) methods [1-6] and failure boundary searching methods [8-9]. However, most of them can only deal with less than 100 independent random variables.

Subset simulation (SUS) [10] is proposed to analyze large-scale circuits. It can estimate the total failure rate by multiplying the failure rates of several intermediate failure events. However, SUS relies on Markov chain MC, and requires a large number of simulation samples to achieve high accuracy.

Multi Failure Region Importance Sampling (MRFIS) [4] extends Importance Sampling methods to high dimension. The key idea is applying sequential quadratic programing (SQP) with multiple starting points (MSP) strategy to find the optimal shift vector (OSV) efficiently. However, the number of samplings is still almost linear to the dimension of problems.

Asymptotic Probability Approximation (APA) [11] and Asymptotic Probability Evaluation (APE) [12] try to cover the correlation among cell-level failures when analyzing the failure rate of a whole circuit with very high dimension. APE decouples the circuit level failure rate into a series of independent local failure events with given global conditions. However, a large number of SPICE simulations are still hard to avoid.

To address the dimensionality issues, a novel and efficient Bayesian yield estimation method is proposed for high dimensional and high sigma SRAM circuits. Considering that the time cost of the low dimensional SPICE simulation is almost neglectable, we try to decrease high dimensional simulations by learning some information from the low-dimensional ones. A Bayesian Gaussian mixture model for performance distribution of high dimensional SRAM column is built by borrowing the prior knowledge from low-dimensional ones. In this approach, SPICE simulations on the high dimensional circuit can be reduced, thus the total run time of algorithm can be saved.

The main contributions of this paper are as follows.

1) We found that the distributions of performance of the low and high dimensional SRAM columns are similar near the failure boundaries. Therefore, we propose a Gaussian mixture model as the likelihood function and encode the prior knowledge as a conjugate prior of Gaussian mixture distribution. Expectation maximization (EM) update rule is applied to obtain the optimal parameter of Gaussian mixture model based on the maximum a posteriori (MAP), where only a few SPICE simulations in high dimension are needed.

2) To improve the efficiency of OSV searching in high dimensional cases, mutual information method between process variations and performance is proposed to reduce dimension.

3) For the proposed methods, the number of simulations on high dimensional SRAM column is nearly $O(1)$ to the bit cell number.

4) Experimental results show 6x speedup over the state-of-the-art method at the 485D cases.

The remainder of this paper is organized as follows. Section 2 simply reviews backgrounds and problem formulation. Then, the proposed method and implementation issues are discussed in Section 3. In Section 4, Experimental results are given to validate our method. Finally, a short conclusion is given in Section 5.

[*] Corresponding authors: yanch@fudan.edu.cn, zhoud@utdallas.edu

DAC '18, June 24–29, 2018, San Francisco, CA, USA
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5700-5/18/06 $15.00
https://doi.org/10.1145/3195970.3195987

2. BACKGROUND

2.1 Problem Formulation

Let $\boldsymbol{x}=[x_1, \ldots, x_D]^T$ be a D-dimensional random variable that denotes all the process parameters, such as threshold voltage and oxide thickness. Without loss of generality, the joint probability density function (PDF) $p(\boldsymbol{x})$ is

$$p(\boldsymbol{x}) = \prod_{d=1}^{D} p_d(x_d), \tag{1}$$

where $p_d(x_d)$ is an independent PDF of process parameters given by foundries, and usually a normal distribution.

SRAM failure rate can be calculated formally as

$$P^{fail} = \int_{\Omega} p(\boldsymbol{x})d\boldsymbol{x} = \int_{-\infty}^{+\infty} I(\boldsymbol{x} \in \Omega) p(\boldsymbol{x}) d\boldsymbol{x}, \tag{2}$$

where Ω denotes the failure region, which is a subset of variation space of process parameters, and $I(\boldsymbol{x} \in \Omega)$ is the indicator function of failure region Ω. If $\boldsymbol{x} \in \Omega$, $I(\boldsymbol{x} \in \Omega)$ is 1, otherwise it is 0.

A SRAM column usually consists of N bit-cells as shown in Figure 1. To read CELL<1>, we pre-charge the bit lines (i.e., BL and BLB) and enable the corresponding word line WL_1, then this bit-cell is selected and connected to the bit lines through the transistors M_3 and M_6. If the differential of bit line voltages (i.e., $\Delta V = V_{BL} - V_{BLB}$) is smaller than the input offset voltage of sense amplifier (i.e., V_{SA_in}) in a given delay, CELL<1> fails.

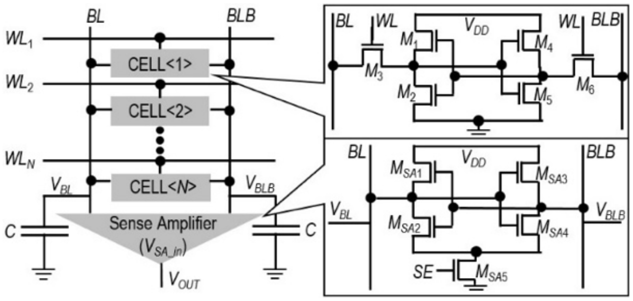

Figure 1. An SRAM column with N bit-cells and a sense amplifier

2.2 Monte Carlo and Importance Sampling Method

Monte Carlo method is the most common and traditional approach to estimate SRAM failure rate. The SRAM failure rate can be probability estimated as

$$\hat{P}_{MC} = \frac{1}{N}\sum_{i=1}^{N} I(\boldsymbol{x}_i), \tag{3}$$

where $\boldsymbol{x}_i$ is the i-th sample.

Sampling efficiency can be improved by IS method. The main idea of IS is replacing the original distribution $p(\boldsymbol{x})$ by *practical distribution* $q(\boldsymbol{x})$. The failure rate then can be expressed as follows

$$P_{IS} = \int \frac{I(\boldsymbol{x})p(\boldsymbol{x})}{q(\boldsymbol{x})} q(\boldsymbol{x}) d\boldsymbol{x}. \tag{4}$$

With N samples generated from $q(\boldsymbol{x})$, failure rate P^{fail} can be estimated as

$$P^{fail} \approx \hat{P}_{IS} = \frac{1}{N}\sum_{i=1}^{N} \frac{I(\boldsymbol{x}_i)p(\boldsymbol{x}_i)}{q(\boldsymbol{x}_i)} \tag{5}$$

However, for a large-scale SRAM column, the CPU time of a single SPICE simulation can be quite large. Figure 2 gives the actual simulation run time of an SRAM column with different number of bit-cells on a 2.67GHz CPU. We can clearly see that the computational complexity of SPICE simulation is $O(n^3)$, where n is the number of bit-cells. Therefore, with the increasing of bit-cell number n, decreasing the samples of high dimensional SRAM column is the key issue.

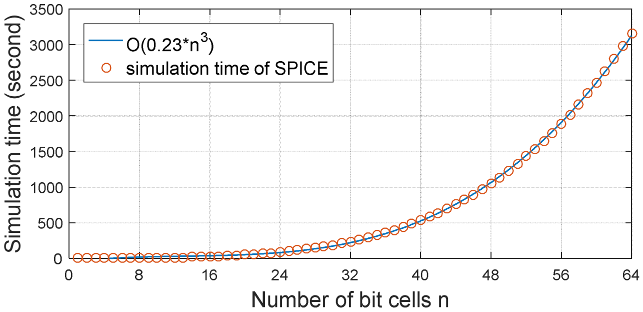

Figure 2. Simulation time versus number of bit-cells in SRAM column

2.3 Gaussian Mixture and Maximum Likelihood Estimate

Distribution of performance denoted by y, depends on process parameters. As a density model, we choose the class of Gaussian mixtures

$$p(y;\boldsymbol{\kappa},\boldsymbol{\mu},\boldsymbol{\sigma}) = \sum_{i=1}^{n} \kappa_i N(y;\mu_i,\sigma_i), \tag{6}$$

where the restrictions $\sum_{i=1}^{n} \kappa_i = 1$ and $\kappa_i \geq 0$ are applied. The $N(y;\mu_i,\sigma_i)$ is multivariate Gaussian distribution with mean μ_i and covariance matrix σ_i.

The Gaussian mixture model is well suited to approximate a wide class of continuous distribution. Based on given data $\{y^k, k = 1, 2, \ldots, m\}$, we can formulate the log likelihood function of Maximum Likelihood Estimate (MLE) as

$$\underset{\boldsymbol{\kappa},\boldsymbol{\mu},\boldsymbol{\sigma}}{Max}\ l(\boldsymbol{\kappa},\boldsymbol{\mu},\boldsymbol{\sigma}) = \underset{\boldsymbol{\kappa},\boldsymbol{\mu},\boldsymbol{\sigma}}{Max} \sum_{k=1}^{m} \log \sum_{i=1}^{n} \kappa_i N(y^k;\mu_i,\sigma_i). \tag{7}$$

Parameter estimations in (7) can be computed with the EM algorithm, which contains the iterative steps of Expectation and Maximization.

In E-step, with current estimate of parameters, the posterior membership probability of pattern y^k is estimated as

$$h_i^k = \frac{\kappa_i N(y^k;\mu_i,\sigma_i)}{\sum_{j=1}^{n} \kappa_j N(y^k;\mu_j,\sigma_j)}. \tag{8}$$

In M-step, we obtain the new parameter estimates

$$\kappa_i = \sum_{k=1}^{m} h_i^k / m \tag{9}$$

$$\mu_i = \sum_{k=1}^{m} h_i^k y^k / \sum_{k=1}^{m} h_i^k \tag{10}$$

$$\sigma_i = \sum_{k=1}^{m} h_i^k (y^k - \mu_i)(y^k - \mu_i)^T / \sum_{k=1}^{m} h_i^k \tag{11}$$

2.4 Bayesian Model for Distribution Estimation

Bayesian can be extended to estimate PDF of performance [13]. Due to process variation, the performance y of circuits can be modeled as a random variable with PDF $p(y)$, which can be expressed as a linear combination of basis functions (e.g. Gaussian, polynomials, etc.) as

$$p(y|\boldsymbol{\theta}) = \sum_{i=1}^{n} w_i b_i(y;v_i)$$
$$\boldsymbol{\theta} = [w_1,...,w_n,v_1,...,v_n] \tag{12}$$

where $b_i(y; v_i)$ is the i-th basis function, the weight factors w_i have the restrictions $\sum_{i=1}^{n} w_i = 1$ and $w_i \geq 0$, n is the total number of basis functions, $\boldsymbol{\theta}$ denotes the parameter vector.

Based on the prior knowledge, the parameter $\boldsymbol{\theta}$ is characterized by a prior distribution as

$$\boldsymbol{\theta} \sim p(\boldsymbol{\theta};\boldsymbol{\omega}) \tag{13}$$

where $\boldsymbol{\omega}$ denotes the hyper-parameters which need to be determined. For calculation simplicity, $p(\boldsymbol{\theta}; \boldsymbol{\omega})$ is often set as the conjugate prior of the likelihood function.

Based on the observations $\boldsymbol{y}=\{y_i, i=1,\ldots,m\}$, we obtain the parameter estimates by maximizing a posterior

$$\hat{\boldsymbol{\theta}} = \underset{\theta}{\text{argmax}}\, p(\boldsymbol{\theta}\,|\,\boldsymbol{y}) = \underset{\theta}{\text{argmax}}\, p(\boldsymbol{\theta})p(\boldsymbol{y}\,|\,\boldsymbol{\theta}), \tag{14}$$

where the likelihood function $p(\boldsymbol{y}|\ \boldsymbol{\theta})$ represents the probability of observing the set $\boldsymbol{y}$ with a given value of $\boldsymbol{\theta}$.

3. PROPOSED APPROACH

Our proposed method contains three components, 1) adopt the concept of mutual information to reduce dimensionality of OSV searching space, 2) build a Gaussian mixture distribution model for the performance distribution of high dimensional SRAM circuit, by borrowing the prior knowledge from a low-dimensional circuit, and 3) calculate the failure rate with an analytical model.

3.1 Efficient OSV Searching Algorithm

Searching OSV is the crucial step in most IS-based method. It is equivalent to solving the following optimization problem

$$\begin{aligned} &min\ \|\boldsymbol{v}\| \\ &s.t.\ \ I(\boldsymbol{v})=1, \end{aligned} \tag{15}$$

where $\boldsymbol{v}$ is in parameter space, $\|\cdot\|$ denotes the L_2-norm. This optimization problem means to find the failure point with the minimal distance to the origin.

MFRIS employs sequential quadratic programming (SQP) [15] as the local search algorithm to find OSV. Although SQP is a reliable method for local search, however, for a D dimensional problem, each step of gradient evaluation needs D+1 times of SPICE simulations, which is almost unaffordable for a problem with hundreds or even thousands of variables.

Note that for an SRAM column, only one bit-cell can be selected at read/write time, thus not all parameter variables are highly relevant to performance. Here, we adopt the concept of mutual information (MI) [16] to reduce dimensionality of parameter variables. MI is a non-negative value, which measures the dependency between the variables. It is equal to zero if and only if two random variables are independent, and larger values mean higher dependency. The MI of two discrete random variables X and Y can be defined as

$$I(\mathrm{X,Y}) = \sum_{y\in Y}\sum_{x\in X} p(x,y)\log(\frac{p(x,y)}{p(x)\mathrm{p(y)}}) \tag{16}$$

where $p(x,y)$ is the joint distribution and $p(x)$, $p(y)$ are the marginal distribution [16].

However, the MI cannot be directly calculated with definition (16). Based on the data set of parameter variables and performances, we use nonparametric methods based on entropy estimation from k-nearest neighbors' distances for fast and accuracy MI estimation [16].

The overall algorithm for OSV searching is described as follows.

Algorithm 1: OSV Searching Algorithm

```
1. Generate samples in parameter space with $n$ times scaled-sigma
2. Simulate these random samples to get performance
3. Use nearest-neighbor method to estimate MI between each variable and performance
4. Select $D'$ variables with highest MI and their summation of MIs accounting for 95% of the total MI
5. Apply SQP to search the OSV in the reduced $D$' dimensional space
```

Experiment data shows that only a few variables have an impact on performance of circuits, thus, we can reduce the dimensionality of OSV searching space tremendously. For example, for SRAM read operation, 10 out of 384 dimensions is accurate enough for OSV searching, thus, we have about 9x speedup compare with the method in [4] even counting the extra 100 SPICE simulations for MI calculations.

3.2 Prior Knowledge Definition

For an SRAM circuit consists of N bit-cells shown in Figure 1 we set the performance as the voltage difference between two bit lines when reading CELL<1>. We try to find the prior knowledge of SRAM column with different number of bit-cells.

Figure 3 shows the performance at the OSV (blue) points and TT corner (red) goes down gradually as the number of bit-cells increases. OSV is obtained in high dimension (384D) by Alg. 1.

In other words, the performance at OSV can be thought as a very smooth function of number of bit-cells. This hints us the possibility of deducing some information from few bit cells to many bit cells. Directly modeling the performance in high dimensional parameter space containing all the parameter variables is still hard. However, instead of the performance itself, the distribution of performance at OSV can concentrate all high dimensional parameter space into a simple PDF. Fortunately, the PDF of performance still varies smoothly with cell number, as shown in Figure 4.

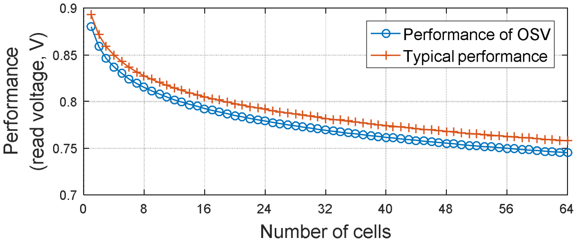

Figure 3. Performance versus the number of bit-cells

Figure 4 shows the performance histogram of SRAM circuits containing 32, 16 and 8 bit-cells respectively. Obviously, these distributions have similar *shape* but different *locations*.

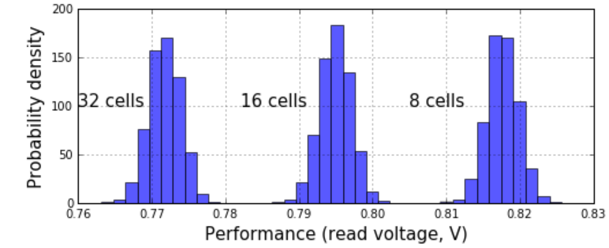

Figure 4. Normed performance histogram of SRAM circuit containing 32, 16 and 8 bit-cells respectively

We assume that the PDF of performance of high dimensional SRAM column is similar to a proper-shifted PDF of performance of the low-dimensional one, but not identical, and the central of the Gauss PDF should be near the OSV. We can encode our prior knowledge so an accurate performance PDF model can be obtained with only a few SPICE simulations on the high dimensional.

For generality, we use the Gaussian mixture distribution to model the PDF of performance. To make sure the posterior distribution have a closed-form expression so that MAP can be easily applied to solve the optimal parameter estimates, we borrow the concept of conjugate prior from Bayesian probability theory. If the posterior distribution is in the same family as the prior distribution, then the prior is called conjugate prior for the likelihood function [14].

The prior of the Gaussian mixture is as follows [14]

$$p(\boldsymbol{\kappa},\boldsymbol{\mu},\boldsymbol{\sigma};\gamma,v,\eta,\alpha,\beta) = D(\kappa;\gamma)\prod_{i=1}^{n} N(\mu_i;v_i,\eta_i^{-1}\sigma_i)W(\sigma_i^{-1};\alpha_i,\beta_i), \tag{17}$$

where Dirichlet distribution $D(\kappa;\gamma)$ is the conjugate prior for the mixture weightings $\kappa = [\kappa_1, \kappa_2, \ldots, \kappa_n]$, and product of an normal distribution $N(\mu_i; v_i, \eta_i^{-1}\sigma_i)$ and a Wishart distribution $W(\sigma_i^{-1}; \alpha_i, \beta_i)$ is the conjugate prior of a single normal distribution [17].

The maximum a posteriori (MAP) parameter estimate in Bayesian approach maximizes the log posterior

$$\begin{aligned} &\underset{\boldsymbol{\kappa},\boldsymbol{\mu},\boldsymbol{\sigma}}{Max}\ l(\boldsymbol{\kappa},\boldsymbol{\mu},\boldsymbol{\sigma}) = \\ &\underset{\boldsymbol{\kappa},\boldsymbol{\mu},\boldsymbol{\sigma}}{Max}\ \sum_{k=1}^{m}\log\sum_{i=1}^{n}\kappa_i N(y^k;\mu_i,\sigma_i) + \log D(\kappa;\gamma) \\ &\quad + \sum_{i=1}^{n}[\log N(\mu_i;v_i,\eta_i) + \log W(\sigma_i^{-1};\alpha_i,\beta_i)]. \end{aligned} \tag{18}$$

With the proper conjugate prior adopted, then EM algorithm derivate in [14] is applied to obtain the optimal parameter estimates. The E-step is identical to (8), and the M-step becomes

$$\kappa_i = (\sum_{k=1}^{m} h_i^k + \gamma_i - 1)/(m + \sum_{i=1}^{n}\gamma_i - n) \tag{19}$$

$$\mu_i = (\sum_{k=1}^{m} h_i^k y^k + \eta_i v_i)/(\sum_{k=1}^{m} h_i^k + \eta_i) \tag{20}$$

$$\sigma_i = \frac{\sum_{k=1}^{m} h_i^k (y^k - \mu_i)^2 + \eta_i(\mu_i - v_i)^2 + 2\beta_i}{\sum_{k=1}^{m} h_i^k + 2\alpha_i - 1}. \tag{21}$$

Once the parameter κ, μ, and σs is solved, the performance distribution can be easily determined by (6), then the failure rate can be calculated with an analytical model which will be mentioned in 3.4. However, the value of hyper-parameter γ, v, η, α and β needs to be set before performing the EM algorithm. We will further discuss the estimation of hyper-parameter in next sub-section.

3.3 Hyper-parameter Estimation

The hyper-parameters are important variables since they control the shape of prior distribution. Suppose we have an additional dataset $\{y^k\ k = 1, 2, \ldots, m'\}$, and we use y_i to denotes the subset of y belongs to i-th Gaussian distribution. Then the hyper-parameter can be estimated with M

$$\begin{aligned} &\alpha_i = \frac{m_i + d}{2},\ \beta_i = \frac{m_i - 1}{2} var(y_i),\ \gamma_i = m_i + 1 \\ &\eta_i = m_i,\ v_i = mean(y_i),\ \text{for i=1, 2, ..., n,} \end{aligned} \tag{22}$$

where $m_i = |y_i|$, $var(\cdot)$ and $mean(\cdot)$ means variance and mean respectively.

However, we cannot simply generate enough additional performance dataset for high dimensional SRAM circuit since it is time-consuming. Besides, even with enough additional dataset y, we cannot determine which Gaussian distribution y^k belongs to. Thus, we obtain the hyper-parameter with enough low-dimensional SRAM circuit performances, and then try to amend these hyper-parameters so they can be even more appropriate for high dimensional SRAM circuit.

$$\begin{aligned} &\alpha_i = \frac{m_i + d}{2},\ \beta_i = \frac{m_i - 1}{2}\sigma_{i,L}^2,\ \gamma_i = m_i + 1 \\ &\eta_i = m_i,\ v_i = \mu_{i,L} + \text{Perf}_H(OSV) - \text{Perf}_L(OSV_L) \\ &\text{for i=1, ..., n,} \end{aligned} \tag{23}$$

where $m_i = \kappa_{i,L} m'$, and $\text{Perf}_H(\cdot)$ and $\text{Perf}_L(\cdot)$ mean the performance of high dimensional circuit and low-dimensional circuit respectively. Since OSV is obtained in high dimension, we discard the components of OSV which do not exist in low-dimensional space to obtain OSV_L. To estimate these hyper-parameters, we first need to build a Gaussian mixture model for performance of low-dimensional circuit with MLE (8)-(11). $\kappa_{i,L}$, $\mu_{i,L}$, $\sigma_{i,L}$ are the parameters of the model.

By substituting (23) into (19)-(21), we can learn an important fact that the hyper-parameters play an important role in determining MAP parameter estimates. If m' is large, MAP parameter estimates for high dimensional circuit is approximately equal to the low-dimensional one, except the shifted parameter μ. In the other extreme case, if m' is sufficiently small, MAP is approximately equal to the MLE estimation.

Algorithm 2: Hyper-parameter Estimation Algorithm

1. Generate samples around OSV in parameter space of low-dimensional SRAM circuit
2. Simulate these random samples with SPICE to get performance
3. Calculate $\kappa_{i,S}$, $\mu_{i,S}$, $\sigma_{i,S}$ with EM-algorithm (8)-(11) derived from MLE
4. Calculate hyper-parameters γ, v, η, α and β with (23)

3.4 Failure Rate Calculation

After performing the MAP mentioned above, we can obtain the PDF of performance near OSV on high dimension cases. However, PDF of performance near the origin cannot be derived directly.

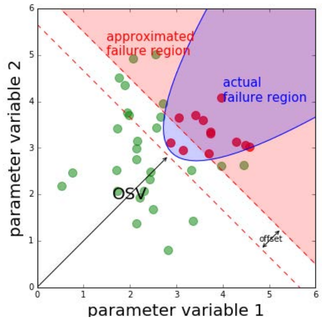

Figure 5. A 2D Schematic diagram of OSV and failure region

Figure 5 shows the difficulty in detail. With the spec of performance and obtained cumulative distribution function (CDF) of performance near OSV, we can easily calculate the failure rate. However, such failure rate is only the weighted area of failure region (i.e., purple filled region in Figure 5) where weight function is the standard norm on OSV. For the failure boundary is unknown and is possibly very complicated to express in high dimension, we cannot obtain the failure rate (i.e., the weighted area of this failure region with the weight of standard norm on origin).

We apply a regression hyper-plane to approximate the complicate high dimensional failure boundary, where the hyper-plane is perpendicular to OSV, and the failure area of regression hyper-plane, i.e., pink filled region in Figure 5, is equal to actual area of fail region under the weight of the standard norm on OSV. Therefore, the offset of the regression hyper-plane to the original OSV normal plane can be analytically calculated as

$$\mathrm{offset} = \mathrm{norminv}\,(P_{OSV}^{success}) \tag{24}$$

where P_{OSV}^{sucess} is the success rate with CDF and *spec* specified, *norminv*$(\cdot)$ is the inverse of the standard normal CDF.

Figure 6 shows the CDF of performance, which is derived from the Gaussian model aforementioned above with a read voltage of a real SRAM column with 32 bit-cells. *spec* is the threshold for distinguishing success and failure. We suppose circuit fails if performance is smaller than the *spec*. If *spec* is given, P_{OSV}^{sucess} can be easily obtained.

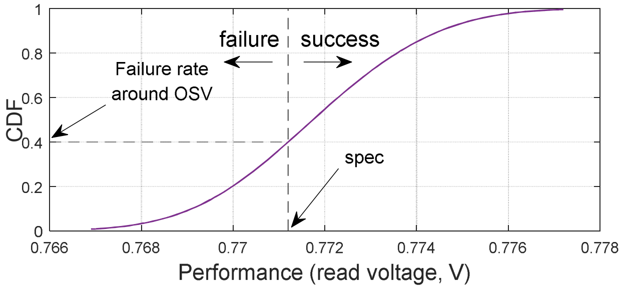

Figure 6. The CDF of performance for high dimensional SRAM

With the offset, i.e., the regression hyper-plane, and assuming it to be the failure boundary, the failure rate at origin can be simply analytically calculated as

$$P_{fail} = 1 - \mathrm{normcdf}\,(\|OSV\| + \mathrm{offset}) \tag{25}$$

where *normcdf*$(\cdot)$ is the standard normal CDF.

3.5 Overview of Algorithm Flow

The overall algorithm is as follows.

Algorithm 3: Bayesian Yield Estimation Method SRAM Circuits

```
1. Sample 100 points in high dimension with 8x sigma, use MI
   for dimensionality reduction, and then apply SQP for OSV
   searching, as shown in Alg. 1
2. Sample 1000 points around OSV in low dimension with
   standard norm, calculating hyper-parameters with (23) as
   shown in Alg. 2
3. Sample 100 points around OSV in high dimension and then
   obtain the Gaussian mixture model of high dimensional
   circuit with EM-algorithm (8) and (19)-(21) based on MAP
4. Calculate failure rate with the analytical equations (25)
```

4. NUMERICAL EXPERIMENTS

To verify the effectiveness and efficiency of the proposed method, the failure rate estimations for read and write operations of an SRAM column is tested. All test cases are under a 28nm CMOS process.

Though SUS [10] can handle the high dimensional cases, it generally needs more than 10^4 samples [4], which is unaffordable for realistic SPICE simulation. We take the results of MNIS with enough samples as the golden results, for its simplicity and clarity on theory. However, MNIS cannot find an accurate OSV within 10^4 samples, so we set the OSV of MNIS as that of MFRIS. Therefore, three different approaches, i.e., MNIS [2], MFRIS [4], and the proposed method, are compared.

Since all methods are intrinsically stochastic, we repeatedly run 10 estimations to acquire the mean and deviation for fair comparisons.

4.1 Failure Rate of Read

For an SRAM read operation, the difference voltage between two bit lines at a given delay is considered as the performance, which should be larger than a given threshold for *success*. To maximize the impact of leakage current, we store 0 in CELL<1> and 1 in all other bit-cells [10].

Table 1 summarizes results of MNIS, MFRIS and proposed method for an SRAM circuit consisting of 32 6T bit-cells and a sense amplifier. Taking the threshold voltages V_{th} of transistors of bit-cells and sense amplifier as parameter variables, thus, we have 197 (32×6+5) random variables in total.

MFRIS can obtain an unbiased estimation of failure rate in high dimensional case with 1327 SPICE simulations and 87.1 hours on a CPU core. The proposed method needs simulations both on one bit-cell column (1×6+5=11D) and 32 bit-cells column (197D). Extra 1000 low dimensional samples are needed for hyper-parameters setting, and the CPU time is 732 seconds, which is ignorable compared with 197D simulation time. 121 samples in 197D are needed for dimensionality reduction and OSV searching, 100 samples for MAP estimations, and totally 221 sample as shown in Table 1. The proposed method is 6.0x faster than the MFRIS with a similar relative errors.

Table 1. Accuracy and Speed Comparisons for read of a 32-bit SRAM column

	Failure Rate	Relative Error	#Samples of 197D	Total CPU time
MNIS	1.26E-06	golden	10644	698.6h/1c*
MFRIS	1.20E-06	7.15%	1327	87.1 h/1c
Proposed	1.20E-06	5.09%	221	14.7 h/1c

* MNIS runs 3.49 hours on 200 cores, 698.6 hours if on a core.

We further verify the proposed method with an SRAM column consisting of 80 bit-cells (80×6+5=485D) as shown in Table 2. Similarly, the proposed method gains 7.7x speedup over MFRIS and again their relative errors are similar.

Table 2. Accuracy and Speed Comparisons for read of an 80-bit SRAM column

	Failure Rate	Relative Error	#Samples of 485D	Total CPU time
MNIS	1.11E-06	golden	6610	2.7days/200c
MFRIS	1.09E-06	6.27%	2360	228h/20c
Proposed	1.08E-06	5.38%	304	29.4h/20c

4.2 Failure Rate of Write Operation

In this sub-section, the write failure of an SRAM array with 80 bit-cells (485D) is estimated. As shown in Figure 1, the initial status of CELL<1> is 1 and we wish to write 0 to it. The write cycle begins by applying the value for the bit lines (i.e., setting *BL* to 0 and *BLB* to 1). *WL₁* is then select and the value is latched in. We consider write node voltage V_{write} of CELL<1> as the performance. If V_{write} is larger than a predefined threshold, it means the circuit fails to write 0 in CELL<1>.

As shown in Table 3, the proposed method gains 6.9x speedup over MFRIS with slight accuracy loss, which is acceptable for engineering application in such high dimensional case.

Note that in SRAM column, write operation mode is very different from read mode, while our proposed Bayesian model is still effective. In fact, for practical SRAM designs, with the increase of bit-cells, the performance will change gradually as shown in Figure 3. Therefore, the failure boundary should move smoothly. Meanwhile, performance is usually nonlinear to parameter variations, but the distribution of performance, i.e., histogram of performance, is usually Gaussian like. Therefore, the assumption of proposed method is that distribution of performance on boundary will move correspondingly with the moving of failure boundary and their shapes of PDF can change subtly. Certainly, if a sudden change of PDF of performance on boundary occurs, the proposed Bayesian model will fail.

Table 3. Accuracy and Speed Comparisons for write of an 80-bit SRAM column

	Failure Rate	Relative Error	#Samples of 485D	Total CPU time
MNIS	1.30E-06	golden	6590	2.6days/200c
MFRIS	1.32E-06	5.39%	2297	213h/20c
Proposed	1.24E-06	7.35%	332	30.8 h/20c

4.3 Computational Complexity vs. Dimensionality

In this subsection, we will verify the computational complexity of proposed method with the increasing dimensionality. Here, we start from an SRAM column with 1 bit-cell (11D) only and add the number of bit-cells to 80 (485D).

Figure 7 shows CPU time, i.e., the number of samples in high dimension of MFRIS and the proposed method. It can see that the computational complexity of MFRIS is roughly linear to dimensionality, while our proposed method is roughly constant. The number of samples goes up slightly because more variables are selected to cover 95% of total MI, which causes the increasing time cost of SQP. For high dimensional SRAM yield analysis, constant time complexity is a very appealing feature.

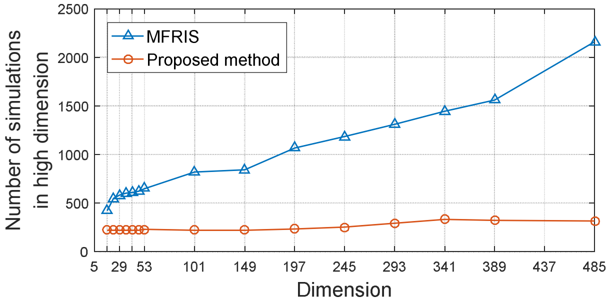

Figure 7. Number of simulations versus dimensionality

5. CONCLUSIONS

In this paper, a novel Bayesian Gaussian mixture model for the performance of high dimensional SRAM circuits is built by borrowing the prior knowledge from low-dimensional ones. EM update rule is applied for obtaining optimal parameters. Besides, mutual information method is proposed to reduce dimension and accelerate the OSV searching. Experimental results verify that the time complexity of proposed method is nearly $O(1)$ to the number of bit-cells, and is 6x speedup over the state-of-the-art method with 485D cases.

6. ACKNOWLEDGEMENTS

This research was supported partly by the National Major Science and Technology Special Project of China (2017ZX01028101-003), partly by National Natural Science Foundation of China (NSFC) research projects 61674042, 61574044, 61574046, 61774045, and 61628402, partly by the Recruitment Program of Global Experts (the Thousand Talents Plan）and partly by NSF Grant 1115564.

7. REFERENCES

[1] 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,” *DAC*, Jul. 2006

[2] Lara Dolecek, Masood Qazi, Devavrat Shah, and Anantha Chandrakasan, “Breaking the simulation barrier: SRAM evaluation through norm minimization,” *ICCAD*, 2008

[3] Qazi, Masood, Tikekar, Mehul, Dolecek, Lara, Shah, Devavrat, Chandrakasan, and Anantha, “Loop flattening & spherical sampling: highly efficient model reduction techniques for SRAM yield analysis,” *DATE*, 2010, pp. 801–806.

[4] Mengshuo Wang, Changhao Yan, Xin Li, Dian Zhou, and Xuan Zeng. “High-Dimensional and Multiple-Failure-Region Importance Sampling for SRAM Yield Analysis,” *IEEE Trans. on VLSI*, vol. 25, no. 3, pp. 806–819, 2017.

[5] C. Dong and X. Li, “Efficient SRAM failure rate prediction via Gibbs sampling,” *DAC*, 2011, pp. 200–205.

[6] Jian Yao, Zuochang Ye, and Yan Wang, “Importance Boundary Sampling for SRAM Yield Analysis With Multiple Failure Regions,” *IEEE Trans. on CAD*, vol. 31, no. 12, pp. 1831–1844, 2011

[7] Solido Design Automation Inc., “High-Sigma Monte Carlo for High Yield and Performance Memory Design” *Solido White Paper*, 2011

[8] Zhenyu Wu, Changhao Yan, Xuan Zeng, and Sheng Guo Wang. “Rapid estimation of the probability of SRAM failure via adaptive multi-level sliding-window statistical method,” *Integration the VLSI Journal*, vol. 50, pp. 1–15, 2015.

[9] Shweta Srivastava and Jaijeet Roychowdhury, “Rapid Estimation of the Probability of SRAM Failure due to MOS threshold Variations,” *IEEE Custom Integrated Circuits Conference*. 229–232, 2007

[10] S. Sun and X. Li, “Fast statistical analysis of rare circuit failure events via subset simulation in high-dimensional variation space,” *ICCAD,* Nov. 2014

[11] H.Yu, Jun Tao, Changhai Liao, *et al*., “Efficient Statistical Analysis for Correlated Rare Failure Events via Asymptotic Probability Approximation,” *ICCAD*, Nov. 2016.

[12] Tao, Jun, Handi Yu, *et al.* "Correlated Rare Failure Analysis via Asymptotic Probability Evaluation." *DAC* 2017:1-6.

[13] X. Li, Wangyang Zhang, Fa Wang *et al*., “Efficient parametric yield estimation of analog/mixed-signal circuits via Bayesian model fusion,” *ICCAD*, pp. 627-634, 2012.

[14] D. Ormoneit and V. Tresp, “Averaging, maximum penalized likelihood and Bayesian estimation for improving Gaussian mixture probability density estimates,”*IEEE Trans. Neural Netw*., vol. 9, no. 4, pp. 639–650, Jul. 1998.

[15] S. Kucherenko and Y. Sytsko, “Application of deterministic lowdiscrepancy sequences in global optimization,” *Comput. Optim. Appl*., vol. 30, no. 3, pp. 297–318, 2005.

[16] Ross, Brian C. "Mutual Information between Discrete and Continuous Data Sets." *Plos One* 9.2(2014):e87357.

[17] J. M. Bernardo and A. F. M. Smith, *Bayesian Theory*. NewYork: Wiley, 1994.