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

ABSTRACT

"Curse of dimensionality" has become the major challenge for existing high-sigma yield analysis methods. In this paper, 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.

CCS CONCEPTS

• **Hardware → Failure prediction**;

KEYWORDS

Process Variation, Failure Probability, Meta-Model, Low-Rank Tensor Approximation

1 INTRODUCTION

As microelectronic devices shrink to nano-meter scale, circuit reliability has become a growing concern due to the uncertainty introduced by process variations. Among various integrated circuits (ICs), memory circuits, such as SRAM bit cells and their peripheral circuits, are typically replicated for millions of times. In this scenario, extremely small circuit failure probability must be considered for a robust circuit design. This high-sigma yield analysis is a "rare-event" problem that deterministic approach is infeasible.

Generally, the probability estimation of "rare-event" is achieved by modern statistical circuit simulation methods. Among these methods, standard Monte Carlo (MC) method remains the golden standard, which repeatedly collects samples and evaluates circuit performance with transistor-level simulations. However, MC is extremely time-consuming for high-sigma case because we need to perform millions of simulations to capture one single failure event.

Prior Work. To avoid expensive MC runs, more efficient approaches have been proposed to sample from the likely-to-fail regions, which can be grouped into two major categories:

(1) Importance Sampling: As a classic modification of MC method, Importance Sampling (IS) attempts to sample from a distorted distribution to capture more failure events. For example, approaches in [1, 2] spherically explore the parametric space and then shift the sampling distribution toward the min-norm point. In order to tackle multi-failure-region circuit cases, methods in [3, 4] try to construct multiple shift vectors and perform mixture importance sampling. More recently, method in [5] develops an iterative resampling approach to search for failure regions and keep the sampling distribution updated. The drawbacks of IS approaches are that the performance is highly dependent on the choice of shift vector, and computational complexity increases exponentially with circuit dimension. None of the aforementioned IS methods can deal with high-dimensional circuit cases.

(2) Classification: Approaches in this category try to filter out the unlikely-to-fail samples and only simulate the remaining samples in failure regions. For example, [6] introduces a linear classifier with safety margin to decrease classification error. Alternatively, [7, 8] utilize conditional classifier and SVM-based nonlinear classifier to handle circuit with multiple failure regions. However, training such classifiers is time-consuming in high dimension and complexity explodes as failure rate decreases.

Paper Contributions. In this paper, 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.

Experiment results validate the accuracy and efficiency of our LRTA approach. For SRAM bit cell with 18 variables, LRTA achieves better accuracy with 6-9X fewer samples when compared to several recent approaches. For 597-dimensional SRAM column with peripheral circuits, LRTA is 2150X faster than MC without compromising accuracy, while other methods fail to converge to ground-truth MC estimation.

2 PRELIMINARIES

2.1 Rare Event Analysis

Suppose that $\boldsymbol{X}$ is a d-dimensional random process variations and its multivariate probability density function (PDF) is defined as $p(\boldsymbol{X})$. Let Y denote the observed performance metric, such as memory read/write time, amplifier gain, etc. Generally, this metric Y requires expensive transistor-level circuit simulations to evaluate.

In statistical circuit simulation scenario, we define the required performance specification as $Y \notin S$, where S is a subset of the entire parametric space. On the contrary, a failure event occurs when $Y \in S$. In general, it is of interest to estimate the probability of Y belonging to subset S. We thereby introduce indicator function $I(\boldsymbol{X})$ to identify pass/fail of Y:

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

Therefore, the probability P_{fail} can be calculated as

$$P_{fail} = P(Y \in S) = \int I(\boldsymbol{X}) \cdot p(\boldsymbol{X}) d\boldsymbol{X} \tag{2}$$

Note that the integral in Eq. (2) is intractable because $I(\boldsymbol{X})$ is unavailable in analytical form. Conventionally, MC method enumerates a sample set $\{\boldsymbol{X}_i\}_{i=1}^{N}$ according to $p(\boldsymbol{X})$ and evaluates their indicator values $\{I(\boldsymbol{X}_i)\}_{i=1}^{N}$ to generate an unbiased estimation of $\hat{P}_{fail}$:

$$\hat{P}_{fail} = \hat{P}(Y \in S) = \frac{1}{N}\sum_{i=1}^{N} I(\boldsymbol{X}_i) \xrightarrow{N \to +\infty} P(Y \in S) \tag{3}$$

2.2 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:

$$\boldsymbol{X} \in \mathbb{R}^d \mapsto \mathcal{M}(\boldsymbol{X}) \in \mathbb{R} \tag{4}$$

Figure 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.

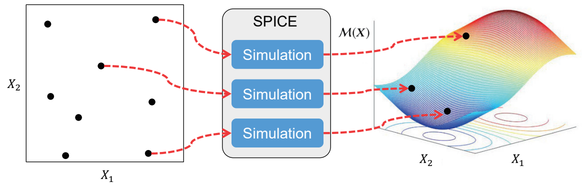


Figure 1: The construction process of meta-model which maps between input variable and output performance

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) [9] 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:

$$\hat{Y} = \mathcal{M}_{PCE}(\boldsymbol{X}) = \prod_{i=1}^{d}\left(\sum_{k=0}^{n_i} \alpha_k^{(i)} \phi_k^{(i)}\right) \tag{5}$$

However, PCE suffers from the problem of "curse of dimensionality". We note that the number of unknown coefficients in Eq. (5) 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.

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:

$$\mathcal{T}_1 = \left\{ v(\boldsymbol{X}) = \left(\otimes_{i=1}^{d} w^{(i)}\right)(\boldsymbol{X}) = \prod_{i=1}^{d} w^{(i)}(X_i); w^{(i)} \in \mathcal{S}^i \right\} \tag{6}$$

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 Eq. (7):

$$\mathcal{S} = \left\{ Y = \sum_{l \in I_n} b_l v_l(\boldsymbol{X}); \quad v_l \in \mathcal{T}_1, b_l \in \mathbb{R} \right\} \tag{7}$$

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*:

$$\hat{Y}^R = \mathcal{M}^R(\boldsymbol{X}) = \sum_{l=1}^{R} b_l v_l(\boldsymbol{X}); \quad v_l \in \mathcal{T}_1, b_l \in \mathbb{R} \tag{8}$$

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

$$v_l(\boldsymbol{X}) = \prod_{i=1}^{d} w_l^{(i)}(X_i) = \prod_{i=1}^{d}\left(\sum_{k=0}^{n_i} z_{k,l}^{(i)} \phi_k^{(i)}(X_i)\right) \tag{9}$$

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 [10]. 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 Eq. (9) into Eq. (8), our proposed LRTA meta-model is formulated as:

$$\hat{Y} = \mathcal{M}^R(\boldsymbol{X}) = \sum_{l=1}^{R} b_l \left(\prod_{i=1}^{d} \left(\sum_{k=0}^{n_i} z_{k,l}^{(i)} \phi_k^{(i)}(X_i) \right) \right) \tag{10}$$

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 1: 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 PROPOSED ADAPTIVE SOLVER

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 $\{b_1, \dots, b_l\}$ 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:

$$\mathcal{R}^{l-1}(\boldsymbol{X}) = Y - \mathcal{M}^{l-1}(\boldsymbol{X}) \tag{11}$$

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

$$v_l = \underset{\gamma \in \mathcal{T}_1}{\operatorname{argmin}} \|\mathcal{R}^{l-1} - \gamma\|^2 \tag{12}$$

Here the solution of Eq. (12) 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:

$$z_l^{(j)} = \underset{\zeta \in \mathbb{R}^{n_j+1}}{\operatorname{argmin}} \left\| \mathcal{R}^{l-1} - \left(\prod_{i \neq j} w_l^{(i)} \right) \left(\sum_{k=0}^{n^j} \zeta_k^{(j)} \phi_k^{(j)} \right) \right\|^2 \tag{13}$$

Eq. (13) 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 $\boldsymbol{b} = \{b_1, \dots, b_l\}$. It is computed by solving a series of minimization problem:

$$\boldsymbol{b} = \underset{\beta \in \mathbb{R}^l}{\operatorname{argmin}} \left\| Y - \sum_{m=1}^{l} \beta_m v_m \right\|^2 \tag{14}$$

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.

4.2 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, Eq. (13) can be improved by adding a ℓ_1-norm constraint $\|\zeta\|_1 \leq \delta$. As ℓ_1-norm is strictly convex, we can equivalently rewrite it as a generalized LASSO [11] problem:

$$z_l^{(j)} = \underset{\zeta \in \mathbb{R}^{n_j+1}}{\operatorname{argmin}} \left\| \mathcal{R}^{l-1} - \left(\prod_{i \neq j} w_l^{(i)} \right) \left(\sum_{k=0}^{n^j} \zeta_k^{(j)} \phi_k^{(j)} \right) \right\|^2 + \lambda \|\zeta\|_1 \tag{15}$$

which is a convex optimization problem. Here λ is a suitable regularization factor. In our implementation, problem (15) is solved by least angle regression algorithm [12], which is a commonly used stagewise procedure for accelerating LASSO problems.

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 Q into three test sets with equivalent samples $\mathcal{U}_i$, $i = \{1, 2, 3\}$. Then training sets consist of corresponding remaining samples $\mathcal{V}_i = Q \backslash \mathcal{U}_i$, $i = \{1, 2, 3\}$.
- Execute our adaptive solver on each training set $\mathcal{V}_i$ with tuned parameters ranging from $1 \le R \le R_{max}$ and $1 \le n_i \le n_{max}$. For each parameter pair $\{R, n_i\}$, we compute their mean square errors $\bar{\epsilon}_{R,n_i}$ comparing with the evaluations from the test sets $\mathcal{U}_i$. Optimal parameter pair $\{R, n_i\}_{op}$ is thereby selected with minimum $\bar{\epsilon}_{R,n_i}$.
- Execute adaptive solver on the whole sample set Q with optimal $\{R, n_i\}_{op}$.

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 Q, and a new sample set Q_c is sequentially added in each iteration by our adaptive sampling framework, which is summarized in Algorithm 2.

Algorithm 2: Adaptive Sampling Algorithm

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

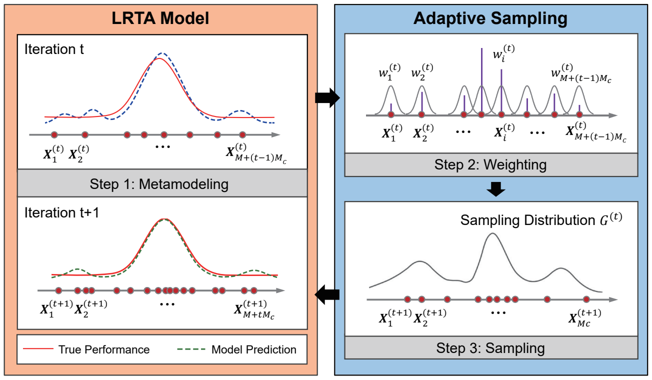

LRTA Model
Iteration t
$X_1^{(t)}$ $X_2^{(t)}$ ⋯ $X_{M+(t-1)M_c}^{(t)}$
Step 1: Metamodeling
Iteration t+1
$X_1^{(t+1)}$ $X_2^{(t+1)}$ ⋯ $X_{M+tM_c}^{(t+1)}$
True Performance
Model Prediction
Adaptive Sampling
$w_1^{(t)}$ $w_2^{(t)}$ $w_i^{(t)}$ $w_{M+(t-1)M_c}^{(t)}$
$X_1^{(t)}$ $X_2^{(t)}$ ⋯ $X_i^{(t)}$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

Figure 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 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.

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) [3] and Adaptive Importance Sampling (AIS) [5] for comparison. The experiment environment is HSPICE with SMIC 40nm model.

5.1 Experiments on 6T SRAM Bit Cell

Figure 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.

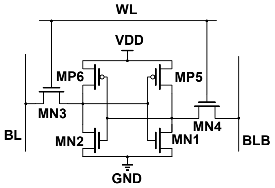
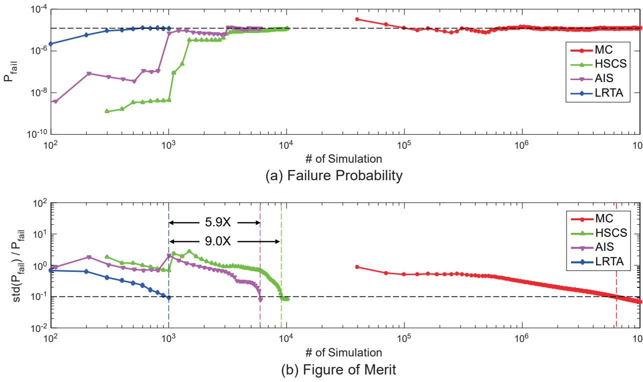


Figure 3: The schematic of typical 6T SRAM cell

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:

$$\rho = \frac{\sqrt{\sigma^2_{\hat{P}_{fail}}}}{\hat{P}_{fail}} \tag{16}$$

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

Table 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(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 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.

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, 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 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.

5.2 Experiments on SRAM Column Circuit

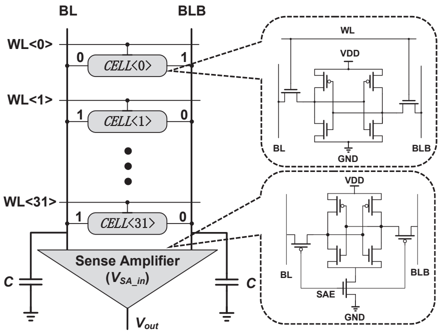
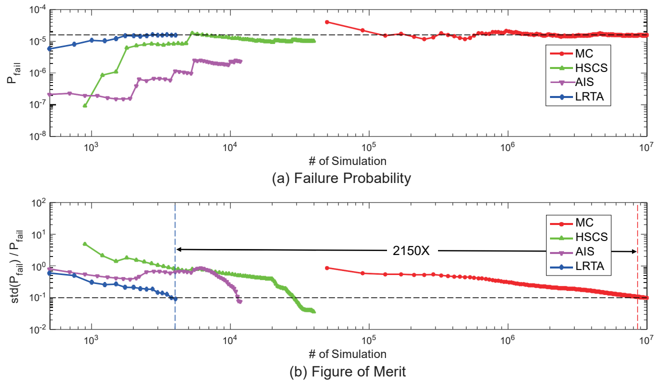


Figure 5: The schematic of 597-dimensional SRAM column with peripheral circuits

A simplified schematic of SRAM column circuit consisting of 32 bit cells and a sense amplifier is shown in Figure 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 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.

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

*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.

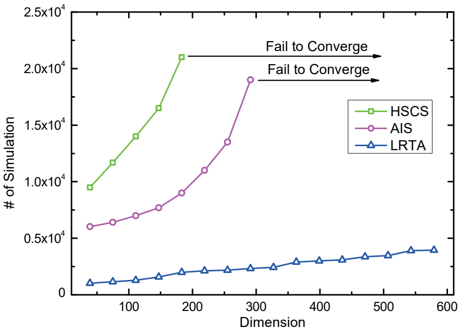


Figure 7: Comparison of required SPICE simulations versus circuit dimension for different methods. HSCS and AIS fail to converge to golden MC estimation as circuits scale up.

Figure 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.

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.

REFERENCES

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

[2] 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 *Proceedings of the Conference on Design, Automation and Test in Europe*, pages 801–806. European Design and Automation Association, 2010.

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

[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 Transactions on Very Large Scale Integration (VLSI) Systems*, 25(3):806–819, 2017.

[5] Xiao Shi, Jun Yang, Fengyuan Liu, 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.

[6] 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 *Design, Automation, and Test in Europe*, pages 235–251. Springer, 2008.

[7] 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 *VLSI Design, 2008. VLSID 2008. 21st International Conference on*, pages 131–136. IEEE, 2008.

[8] 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. ACM, 2014.

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

[10] 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.

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

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