Seeking the Yield Barrier: High-Dimensional SRAM Evaluation Through Optimal Manifold

Yanfang Liu†
School of Integrated Circuit Science and Engineering, Beihang University
Beijing, China
liuyanfang@buaa.edu.cn

Guohao Dai†
College of Mechatronics and Control Engineering, Shenzhen University
Shenzhen, China
daiguohao2019@email.szu.edu.cn

Wei W. Xing*
School of Integrated Circuit Science and Engineering, Beihang University
Beijing, China
wxing@buaa.edu.cn

2023 60th ACM/IEEE Design Automation Conference (DAC) | 979-8-3503-2348-1/23/$31.00 ©2023 IEEE | DOI: 10.1109/DAC56929.2023.10247952

***Abstract*—Being able to efficiently obtain an accurate estimate of the failure probability of SRAM components has become a central issue as model circuits shrink their scale to submicrometer with advanced technology nodes. In this work, we revisit the classic norm minimization method. We then generalize it with infinite components and derive the novel optimal manifold concept, which bridges the surrogate-based and importance sampling (IS) yield estimation methods. We then derive a sub-optimal manifold, optimal hypersphere, which leads to an efficient sampling method being aware of the failure boundary called onion sampling. Finally, we use a neural coupling flow (which learns from samples like a surrogate model) as the IS proposal distribution. These combinations give rise to a novel yield estimation method, named Optimal Manifold Important Sampling (OPTIMIS), which keeps the advantages of the surrogate and IS methods to deliver state-of-the-art performance with robustness and consistency, with up to 3.5x in efficiency and 3x in accuracy over the best of SOTA methods in High-dimensional SRAM evaluation.**

***Index Terms*—Yield Analysis, Importance Sampling, Normalization Flow**

I. Introduction

As the technology of integrated circuits develops, microelectronic devices shrink their scale to submicrometer, which makes random process variations, e.g., intra-die mismatches, doping fluctuation, and threshold voltage variation, crucial factors to be considered in a circuit design. The situation gets worse in modern circuit designs, where some cells can be replicated millions of times in a circuit, e.g., in an SRAM cell array. A cornerstone to resolving the increasing concern of yield is the development of efficient yield estimation methods, which provide an accurate and fast failure probability estimation for a given circuit design under specific process variations.

Monte Carlo (MC) is the golden standard baseline, and it is commonly utilized to estimate the yield across industry and academia. In a nutshell, MC runs SPICEs (Simulation Program with Integrated Circuit Emphasis) with parameters sampled from the process variation distribution millions of times and counts the number of failures to deliver accurate estimation. Obviously, MC is computationally expensive and easily becomes infeasible for problems for low-yield problems, which is rather common in modern circuit designs, e.g., the yield of a 65nm SRAM cell array can be 10^{-5}.

To improve the efficiency of yield estimation, importance-sampling (IS)-based methods have been proposed. Instead of drawing samples from the default normal distribution, IS methods draw samples from a proposal distribution, which should be designed to approximate the oracle failure distribution. Thus, most efforts tried to design a proposal distribution that can approximate the failure distribution well. For instance, [1] shifts the sampling centroid of the normal distribution to the closest failure point as the proposal distribution, which is well-known as norm minimization (NM). Based on the shifting idea, [2] proposes to sample from multiple failure regions using a hyperspherical clustering method. Instead of relying on a static proposal distribution, [3] proposes an adaptive importance sampling (AIS) to update the shifted distribution as more samples are collected. To better fit the failure distribution in a high-dimensional space, [4] samples from multiple regions clustered by multi-cone clustering and sequentially updates its proposal distribution. AIS is further enhanced by [5] by introducing a mixture of von Mises-Fisher distributions to replace the standard normal distribution. The IS methods are robust and simple to implement, making them popular in the industry. Nonetheless, they still require a large number of SPICE simulations and can not incorporate coming knowledge of new samples to update their models (e.g., the proposal distribution and/or its family).

Another main path to efficient yield estimation is utilizing powerful machine learning (ML) to build a data-driven surrogate model to approximate the unknown indicator function and use active learning to sequentially reduce prediction error. [6] uses a Gaussian process (GP) to approximate the underlying performance function and an entropy reduction method for active learning. Based on the same updating scheme, [7] replaces the GP with a nonlinear-correlated deep kernel method with feature selection to identify the determinant features to focus on. Instead of using a two-stage approach that potentially introduces bias, [8] uses a low-rank tensor approximation to the polynomial chaos expansion (PCE) to approximate the performance function. Deep learning (e.g., RBF network) can (also be) utilized in combination with importance sampling to compute the yield [9]. Despite their success, the surrogate methods inevitably suffer from the "curse of dimensionality". More specifically, the high dimensionality (which is quite common in SRAM circuits) makes it challenging to compute the integration over the domain and data demanding to train the surrogate model, which can defeat the purpose of introducing a surrogate model. Another critical problem with surrogate-based methods is the highly nonlinear optimization problem involved in the surrogate model training, which, if not done right, can lead to a wrong

This work is supported by Fundamental Research Funds for the Central Universities; experiments are supported by Primarius Technologies Co.,Ltd.
†Both authors contributed equally to this research.
*Corresponding author.

surrogate model and thus a wrong yield estimation, a disaster the industry can not afford.

In this work, we aim to combine the advantages of IS and surrogate methods to deliver an efficient and, most importantly, robust yield estimation. To this end, we first revisit the classic NM method (based on which many methods are proposed); we generalize it with infinite components and derive the optimal manifold, which reveals the close connection between IS and surrogate methods and serves as a guideline in designing the proposal distribution in IS methods. Based on the optimal manifold, we propose a sub-optimal solution, optimal hypersphere, and derive onion sampling, which provides efficient and robust samples from the failure distribution. Finally, we introduce neural spline flows (NSF) as the proposal distribution; it sequentially approximates the truth failure probability with more samples collected (as in a surrogate method) to deliver efficient sampling. This combination gives rise to a novel sampling method, named **Opti**mal **M**anifold **I**mportant **S**ampling (OPTIMIS), which absorbs the advantages of surrogate-based and IS-based methods to deliver state-of-the-art (SOTA) performance with robustness and efficiency. The novelty of this work includes:

1) Optimal manifold: a generalization of the classic NM.
2) Onion sampling, which can efficiently sample from the failure distribution
3) OPTIMIS, combining onion sampling and NSF to deliver SOTA yield estimation; which is as efficient as the surrogate methods and as robust as the IS methods.
4) The superiority is valid with SRAM circuits with up to 1093 dimensions, ablation study, and robustness test.

II. Background

A. Problem Definition

Denote $\mathbf{x} = [x^{(1)}, x^{(2)}, \cdots, x^{(D)}]^T \in \mathcal{X}$ as the variation process parameter, and $\mathcal{X}$ the variation parameter space. $\mathcal{X}$ is generally a high-dimensional space (i.e., large D); Each variable in $\mathbf{x}$ denotes the variation parameters of a circuit during manufacturing, e.g., length or width of PMOS and PNOS transistors. In general, $\mathbf{x}$ are considered mutually independent Gaussian distributed, $p(\mathbf{x}) = (2\pi)^{\frac{D}{2}} \exp\left(-||\mathbf{x}||^2/2\right)$. Given a specific value of $\mathbf{x}$, we can evaluate the circuit performance $\mathbf{y}$ (e.g., memory read/write time and amplifier gain) through SPICE simulation, $\mathbf{y} = \mathbf{f}(\mathbf{x})$, where $\mathbf{f}(\cdot)$ is the SPICE simulator, which is considered an expensive and time-consuming black-box function; $\mathbf{y} = [y^{(1)}, y^{(2)}, \cdots, y^{(K)}]^T$ are the collections of circuit performance based on the simulations. When K metrics are all smaller than or equal to their respective thresholds (predefined by designers) $\boldsymbol{t}$, i.e., $y^{(k)} \leq t^{(k)}$ for $k = 1, \cdots, K$, the circuit is considered as a successful design; otherwise, it is a failure one. We use failure indicator $I(\mathbf{x})$, which is 1 if $\mathbf{x}$ lead to a failure design and 0 otherwise, to denote the failure status of a circuit. Finally, the ground-truth failure rate $\hat{P}_f$ is: $\hat{P}_f = \int_{\mathcal{X}} I(\mathbf{x})\, p(\mathbf{x}) d\mathbf{x}$.

B. Monte Carlo and Importance Sampling

The direct calculation of the yield is intractable due to the unknown $I(\mathbf{x})$. A common approach to estimate the failure probability is Monte Carlo (MC), which is easily implemented by sampling $\mathbf{x}_i$ from $p(\mathbf{x})$ and evaluating the failure probability by the ratio of failure samples to total samples. More specifically, $\hat{P}_f$ is approximated by $P_f = \frac{1}{N}\sum_{i=1}^{N} I(\mathbf{x}_i)$, where $\mathbf{x}_i$ is the i-th sample from $p(\mathbf{x})$, and N is the number of samples. When $N \to \infty$, $P_f \to \hat{P}_f$. To obtain an estimate of $1-\varepsilon$ accuracy with $1-\delta$ confidence, $N \approx \frac{\log(1/\delta)}{\varepsilon^2 \hat{P}_f}$ is required. For a modest 90% accuracy ($\varepsilon = 0.1$) with 90% confidence ($\delta = 0.1$), we need $N \approx 100/\hat{P}_f$ samples, which is infeasible in practice for small $\hat{P}_f$, says, 10^{-5}. We can also see this intuitively from the fact that it requires averagely $1/\hat{P}_f$ samples just to observe a failure event.

Instead of drawing samples from $p(\mathbf{x})$, the IS methods draw samples from a proposal distribution $q(\mathbf{x})$ and estimate

$$P_f = \int_{\mathcal{X}} \frac{I(\mathbf{x})p(\mathbf{x})}{q(\mathbf{x})} q(\mathbf{x}) d\mathbf{x} \approx \frac{1}{N}\sum_{i=1}^{N} \frac{I(\mathbf{x}_i)p(\mathbf{x}_i)}{q(\mathbf{x}_i)}, \tag{1}$$

where $\mathbf{x}_i$ are samples drawn from $q(\mathbf{x})$ and are used to approximate the integral as in MC. For convenience, we define the importance weight $w(\mathbf{x}) = \frac{p(\mathbf{x})}{q(\mathbf{x})}$. Eq. (1) is more efficient than MC if $q(\mathbf{x})$ is chosen carefully.

C. Norm Minimization

One of the foundation work in IS for yield is the Norm Minimization (NM [1], aka *optimal shift vector*), which samples from a normal distribution centered at $\boldsymbol{\mu}^*$,

$$\boldsymbol{\mu}^* = \operatorname{argmin} ||\mathbf{x}||^2 \quad \text{s.t.} \quad I(\mathbf{x}) = 1, \tag{2}$$

where $||\mathbf{x}||^2 = \sum_{d=1}^{D} (x^{(d)})^2$ is the Euclidean norm.

III. Proposed Approach

OPTIMIS relies on two key components: onion sampling and NSF, both of which are based on the optimal manifold generalized from NM. A heads-up example of OPTIMIS to approximate five toy failure probability distributions using 1000 samples is shown in Figure 1 to show the effectiveness.

A. Optimal Proposal Distribution in IS

From Eq. (1), we can see that the optimal proposal distribution $q(\mathbf{x})$ is the one that minimizes the approximate variance given by the Delta method, i.e.,

$$q^*(\mathbf{x}) = \operatorname*{argmin}_q \mathbb{E}_q \left[w^2(\mathbf{x}) \left(I(\mathbf{x}) - \hat{P}_f \right)^2 \right]. \tag{3}$$

Utilizing Lagrange multiplier rule for calculus of variations, we can show that the optimal proposal distribution is given by

$$q^*(\mathbf{x}) = p(\mathbf{x}) I(\mathbf{x}) / \hat{P}_f. \tag{4}$$

If we take a Laplace approximation, we have

$$q^*(\mathbf{x}) \approx p(\mathbf{x}) I(\mathbf{x}) \exp\left(-(\mathbf{x}-\hat{\boldsymbol{\mu}})^T \mathbf{S}^{-1} (\mathbf{x}-\hat{\boldsymbol{\mu}})\right) / \hat{P}_f, \tag{5}$$

where $\hat{\boldsymbol{\mu}} = \operatorname{argmax}_{\mathbf{x}} \log p(\mathbf{x}) I(\mathbf{x}); \mathbf{S}^{-1} = \nabla^2_{\mathbf{x}} \log p(\hat{\boldsymbol{\mu}}) I(\hat{\boldsymbol{\mu}})$. Notice that $p(\mathbf{x})$ is just a Gaussian, which is monotonically decreasing in $|\mathbf{x}|$. Thus, $\hat{\boldsymbol{\mu}}$ is obtained for the smallest $\mathbf{x}$ that satisfies $I(\mathbf{x}) = 1$, exactly the solution in NM of Eq. (2). Furthermore, the Laplace approximation indicates how to design the covariance properly. However, $I(\mathbf{x})$, in this case, is not a continuous function, and the computation of $\mathbf{S}^{-1}$ is ill-posed. That is probably the reason most previous works use a preset diagonal covariance [1], [3], leading to inferior performance due to the ignorance of the variance, which we will discuss later.

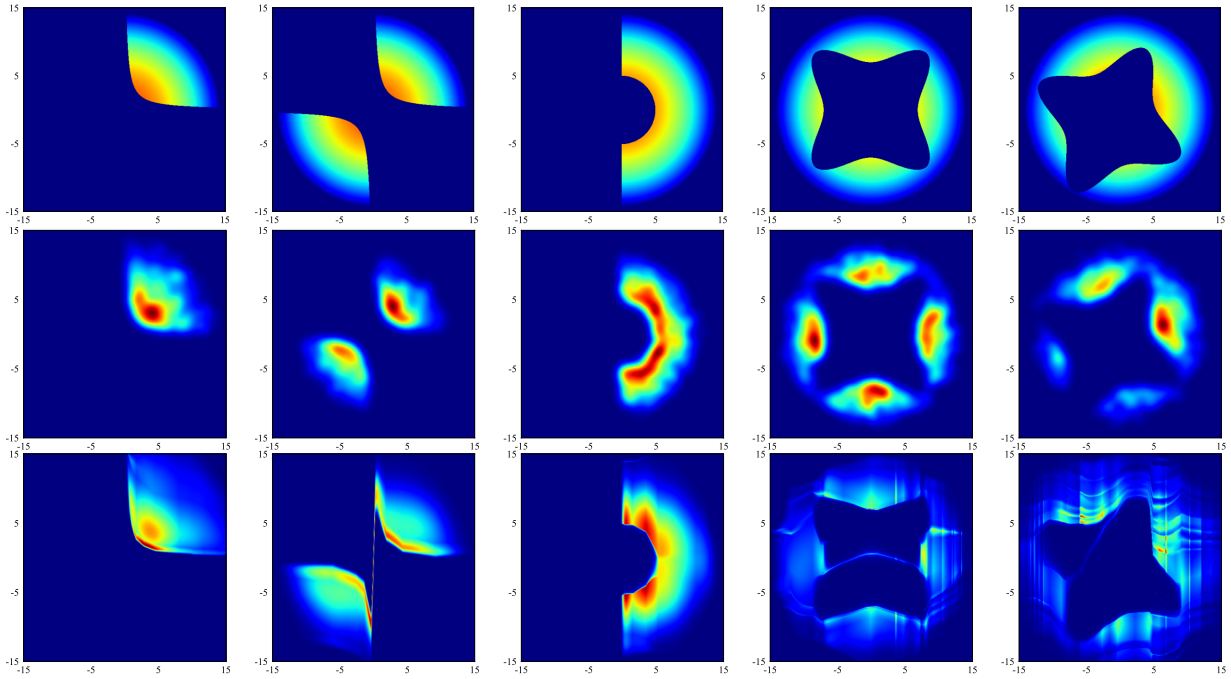

Fig. 1. Illustration of OPTIMIS in five (columns) 2D toy examples. Top row: the true log failure probability. Middle row: the log failure probability estimated with KDE based on onion sampling with 1000 samples. Bottom row: the log failure probability estimated by NF based on the onion samples.

However, the main issue of NM is that it seeks only the closest single failure region and ignores other failure regions, leading to inferior performance. For instance, in Fig. 1, NM will only work for the first case as the other four cases have multiple failure regions. To resolve this issue, the SOTA methods seek a mixture of distributions (e.g., vMF [5]) to approximate the optimal proposal distribution. However, the number of mixture components is highly problem-dependent, making this approach impractical for real problems.

B. Optimal Manifold

To enhance NM for more complex problems, let us equip it with an infinite mixture of Gaussian to drastically improve its model capacity $q(\mathbf{x}) = \sum_{i=1}^{M} w_i \mathcal{N}(\mathbf{x} - \boldsymbol{\mu}_i, s\mathbf{I})$, where w_i is the weight and $\mathbf{M} \to \infty$. Unlike the Laplace approximation, to avoid ignorance of the variance, we minimize the KL divergence

$$\mathrm{KL}(q^*(\mathbf{x})||q(\mathbf{x})) = \mathbb{E}_{q^*(\mathbf{x})}\left[\log q^*(\mathbf{x})\right] - \mathbb{E}_{q^*(\mathbf{x})}\left[\log q(\mathbf{x})\right], \quad (6)$$

where $\mathbb{E}_{q^*(\mathbf{x})}\left[\log q^*(\mathbf{x})\right]$ is a constant of the entropy of the optimal proposal distribution. Minimization of the KL divergence is equivalent to maximizing $\mathbb{E}_{q^*(\mathbf{x})}\left[\log q(\mathbf{x})\right]$, which is the low bound of the optimal solution. We now aim to optimize

$$\underset{\{\boldsymbol{\mu}_i, w_i\}_{i=1}^{M}}{\operatorname{argmax}} \int p(\mathbf{x}) I(\mathbf{x}) / \hat{P}_f \log \left(\sum_{i=1}^{M} w_i \mathcal{N}(\mathbf{x} - \boldsymbol{\mu}_i, s\mathbf{I}) \right) \mathrm{d}\mathbf{x}. \quad (7)$$

The complete solution to Eq. (7) might seem complicated. But we can see that to get the maximum, the main volumes of the Gaussian components (corresponding to a large w_i) should be placed near the failure boundaries $B(\mathbf{x}) = \partial I(\mathbf{x}) / \partial x \neq 0$. We call this the optimal manifold, which does not necessarily equal $B(\mathbf{x})$. In practice, we cannot work with a Gaussian mixture with infinite components. The optimal manifold suggests a suboptimal solution to Eq. (7) with a finite number of Gaussian components. If we set $M = 1$, the optimal manifold becomes a variational version of NM, which should provide better performance than the ordinary NM. Nevertheless, even the variational NM is difficult to seek because $I(\mathbf{x})$ is unknown, which really gives credits to NM for its practicality.

The crucial point to get the optimal manifold or the optimal proposal distribution is to approximate $I(\mathbf{x})$. In the work of [6], the authors derive a general framework for active learning with optimal proposal samples, which are precisely the points that reduce the uncertainty of $I(\mathbf{x})$. This gives us a direct connection between IS and surrogate methods for that they actually share the same intermediate step, approximating $I(\mathbf{x})$, in order to efficiently estimate $\hat{P}_f$. The difference is that most IS methods update the information about $I(\mathbf{x})$ implicitly by random samples to reveal the unseen $B(\mathbf{x})$. Because the samples are generated randomly, this procedure is not target orientated and thus is not as efficient, which also comes with the advantage that it will not be trapped in local minima and will always converge. On the other hand, surrogate methods update the information about $I(\mathbf{x})$ explicitly by reducing the uncertainty of $I(\mathbf{x})$. Because $I(\mathbf{x})$ is approximated with a regression that is highly non-convex to optimize, and the optimization of the proposal points is again highly complex, this procedure will be trapped in local minima and will not always converge.

To combine both advantages of IS and surrogate methods, we can use a complex generative model, says, normalizing flows (NF), to learn from data and approximate $q^*(\mathbf{x})$, while the computation of $\hat{P}_f$ is still done by IS with samples drawn from the NF. This way, even if the proposal distribution is suboptimal, the IS methods will always converge. In the meanwhile, the NF can use existing samples to implicitly learn the target failure probability. Although NF is a powerful model, it contains a deep neural network (NN) and is thus data-demanding. The challenge left is how to generate as many failure points as possible with limited resources while these samples must come from $q^*(\mathbf{x})$.

C. Optimal Hypersphere

Before we move to NF and the optimal manifold, we will derive the optimal hypersphere and an effective onion sampling to provide training data for NF. As discussed, the failure

boundary $B(\mathbf{x})$ is unknown, and there is no way to actually solve the optimization in Eq. (7) in practice. We thus turn to an easier solution by constraining the Gaussian centroids lying on a hypersphere of radius r, that is $\boldsymbol{\mu}_i^2 = r$. We aim to solve

$$\underset{\{\boldsymbol{\mu}_i, w_i\}_{i=1}^M}{\operatorname{argmax}} \int p(\mathbf{x}) I(\mathbf{x}) / \hat{P}_f \log \left(\sum_{i=1}^{M} w_i \mathcal{N}(\mathbf{x} - \boldsymbol{\mu}_i, s\mathbf{I}) \right) \mathrm{d}\mathbf{x}, \quad (8)$$

s.t. $||\boldsymbol{\mu}_i||^2 = r$. With $M \to \infty$, solving (8) is equivalent to optimizing the radius t such that the integral over $I(\mathbf{x})p(\mathbf{x})$ is maximized. We call such a hypersphere with radius r the optimal hypersphere.

Certainly, the optimal hypersphere is also intractable due to the absence of $I(\mathbf{x})$. However, it shows us that the maximum is achieved by a hollow hypersphere, with its radius r being near the failure boundary $B(\mathbf{x})$ with the main volume.

The optimal hypersphere suggests that we can sample inside a hollow hypersphere to efficiently generate samples that are both likely to fail ($I(\mathbf{x}) = 1$), and come from the original parameter distribution $p(\mathbf{x})$. We propose a novel onion sampling inspired by the optimal hypersphere in this section.

For a hypersphere with radius r, we can compute the cumulative distribution function (CDF) i.e., $F(r) = \int p(|\mathbf{x}| < r)\mathrm{d}\mathbf{x}$. Following the Latin hypercube sampling (LHS), we divide the domain with K hollow hyperspheres such that $F(r_k) = \frac{k}{K}$, where r_k is the radius of the k-th hypersphere. The particular value of r_k is easy to compute because the inverse of $F(r)$ can be computed analytically. If we select each hypersphere with probability $1/K$ and generate J samples inside the hypersphere using a uniform distribution (which will allow us to effectively explore the domain for failure regions), we can reproduce sampling from $p(\mathbf{x})$ with precision proportional to $1/K$.

As the optimal hypersphere suggests, we should avoid sampling in the center, which is likely to be a non-failure region for the yield problem. Instead of choosing a hypersphere randomly, we start from the largest hypersphere $r = r_K$. Inside the sphere, we sample J points uniformly and put them through the SPCIE, compute $I(\mathbf{x})$, and keep all failure samples. We also define the failure rate under the uniform sampling for the k-hypersphere as U_k, which gives us an indication when we approach the failure boundary $B(\mathbf{x})$ where U_k will experience a sudden drop. We repeat this process until U_k is below a threshold τ. A smaller τ results in a more accurate sampling but will also increase the computational cost. This sampling process is like peeling an onion layer by layer, and thus we call it onion sampling, which is summarized in Algorithm 1.

The onion sampling can be further improved for practical use. We discuss some scenarios here. As discussed in the optimal sphere, to have a good approximation of $q^*(\mathbf{x})$, the key is to have a good matching of the main volume for $q^*(\mathbf{x})$. If we have more but a limited budget left for the pre-sampling stage, we can repeat the previous procedure but start from the ending hypersphere near the optimal hypersphere going outward. If the K-th hypersphere is not reached during this process, the total samples might be a bit distorted from $\mathbf{q}^*(\mathbf{x})$ but still provide a good training set for NF, which will correct such distortion during its update iterations. If there is more budget, we might exclude the possible non-failure regions, re-divide the domain into K hyperspheres, and then repeat the onion sampling process. This will give us a closer approximation to the optimal hypersphere.

Algorithm 1 Onion Sampling

Require: SPICE-based Indication $I(\mathbf{x})$, # of hypersphere K, number of # per hypersphere J, threshold τ

1: Divide the domain into K hyperspheres with equally increased cumulative probability
2: **while** $U_k > \tau$ **do**
3: Uniformly generate J points inside k hypersphere
4: Keep samples that fails, $\mathbf{X} = \mathbf{X} \cup \mathbf{x}_{kj}$ for $I(\mathbf{x}_{kj}) = 1$
5: Compute uniform failure rate U_k; $k = k - 1$
6: **end while**
7: **return** Failure sample collections $\mathbf{X}$

D. Normalizing Flows For Optimal Proposal Distribution

The onion sampling is a simple yet effective method for sampling approximately from the optimal proposal distribution $q^*(\mathbf{x})$ relying on the optimal hypersphere, which is a suboptimal solution based on the optimal manifold. With samples from onion sampling, we now harness the power of modern deep learning and massively parallel computing to approximate the optimal manifold and further improve our performance. More specifically, we implement an NF with Neural Spline Flows (NSF) [10].

NF is a class of generative models that can approximate any complex distributions by transforming a simple base distribution using a series of invertible transformations. In our case, the base distribution is naturally the standard normal $p(\mathbf{x})$ whereas the target distribution $q^*(\mathbf{x})$. Assume a mapping $\mathbf{x} = \mathbf{g}(\mathbf{z})$, where $\mathbf{g} : R^D \to R^D$ is a bijective function, and $\mathbf{z} \in R^D$ is a random vector drawn from a normal distribution $p(\mathbf{z})$ (which distinguishes itself form $p(\mathbf{x})$). The PDF $q(\mathbf{x})$ can be expressed using the change of variables formula:

$$q(\mathbf{x}) = p(\mathbf{h}(\mathbf{x}))|\det D\mathbf{h}(\mathbf{x})| = p(\mathbf{z})|\det D\mathbf{g}(h(\mathbf{x}))|^{-1}, \quad (9)$$

where $\mathbf{h}(\cdot)$ is the inverse function of $\mathbf{g}(\cdot)$; $D\mathbf{h}(\mathbf{x}) = \frac{\partial \mathbf{h}(\mathbf{x})}{\partial \mathbf{x}}$ is the Jacobian matrix of $\mathbf{h}(\mathbf{x})$; likewise, $D\mathbf{g}(\mathbf{z}) = \frac{\partial \mathbf{g}(\mathbf{z})}{\partial \mathbf{z}}$ is the Jacobian matrix of $\mathbf{g}(\mathbf{z})$. Sampling from $q(\mathbf{x})$ is equivalent to sampling from $p(\mathbf{z})$ and then applying mapping $\mathbf{x} = f(\mathbf{z})$. The key to delivering a close approximation to $q^*(\mathbf{x})$ is to choose a proper mapping $\mathbf{g}(\cdot)$, which admits an efficient inversion and Jacobian matrix computation.

After many trial tests with different models (including autoregressive flow, affine coupling flow, planar flows, etc.), we found NSF [10] works the best for the yield problem. In NSF, $\mathbf{z}$ is divided into two parts, $\mathbf{z} = (\mathbf{z}^{(1:d)}, \mathbf{z}^{(d+1:D)})^T$, and

$$\mathbf{x}^{(1:d)} = \mathbf{g}_\phi(\mathbf{z}^{(1:d)}), \quad \mathbf{x}^{(d+1:D)} = \mathbf{g}_\theta(\mathbf{z}^{(d+1:D)}), \quad (10)$$

where $\boldsymbol{\theta}(\mathbf{z}^{(1:d)})$ is a NN that takes $\mathbf{z}^{(1:d)}$ as input and outputs the model parameters for a spline mapping $\mathbf{g}_\theta(\mathbf{z}^{(d+1:D)})$. $\mathbf{g}_\theta(\cdot)$ is a monotonic rational-quadratic spline whose each bin is defined to be the ratio of two rational-quadratic polynomials [10].

Training the NF is straightforward by maximum likelihood estimation (MLE), $\mathcal{L} = \sum_{i=1}^{N} \log q(\mathbf{x}_i)$, using stochastic gradient descent. The gradient of the log-likelihood is easily computed via modern automatic differentiation tools based on chain rules.

NSF offers an excellent combination of functional flexibility whilst maintaining a numerically stable inverse that is of the same computational and space complexities as the forward

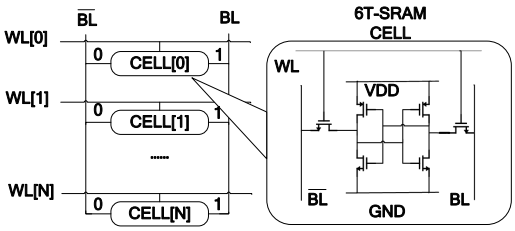

Fig. 2. The structure of SRAM column circuit

operation. This element-wise transform permits the accurate representation of complex multivariate distributions. Another advantage of NSF is that it can deal with high-dimensional problems through the flow by identifying the most important dimensions and then focusing on them.

IV. Experimental Results

We firstly examine OPTIMIS in five toy examples with different artificial failure boundaries (e.g., open boundaries, multiple failure regions, and non-centered regions) with their ground-truth log failure probability (LFP) in Fig. 1. Onion sampling with 1000 samples and estimated LFP using kernel density estimator (KDE) with a bandwidth of 0.75 are shown in the second row. We can see that onion sampling is efficient but also overestimate the LFP. NSF for the estimated LFP is shown in the third row, which shows significant reductions in the overestimation of onion sampling.

We assess OPTIMIS on challenging high-dimensional benchmark circuits, namely, three SRAM column circuits with 108, 569, and 1093 variation parameters, respectively. The circuits are synthesized using the Synopsys Design Compiler and the Cadence Virtuoso design tools. We set the target failure rate of the circuits at approximately 10^{-5} to highlight the challenge of the yield estimation problem. We implement MC as the golden standard to estimate the ground-truth yields. To show the accuracy and efficiency of OPTIMIS, we also implement the SOTA IS methods, including Minimized Norm Importance Sampling (MNIS) [1], Hyperspherical Clustering and Sampling (HSCS) [2], Adaptive Importance Sampling (AIS) [3], Adaptive Clustering and Sampling (ACS) [4], and surrogate-based methods, including Low-Rank Tensor Approximation (LRTA) [8] and Absolute Shrinkage Deep Kernel learning (ASDK) [7], as comparison methods.

The figure of Merit (FOM) $\rho = \mathrm{std}(P_f)/P_f$ (where $\mathrm{std}(P_f)$ is the stand deviation of estimated yield) is used as the stopping criterion for all methods with $\rho = 0.1$ (indicating at least 90% accurate with 90% confidence interval) as in many previous works, e.g., [1], [2], [9]. For the 108-dimensional case, we use a 4-layer multi-layer perception (MLP), each with 432 hidden units; for the high-dimensional 569 and 1093 cases, we use a 7-layer MLP with 600 hidden units for each layer. ReLu activation is used. The optimization is done with Adam with 500 epochs. The baseline methods use (default) setting as suggested in their papers. All experiments are performed on a Linux system with AMD 5950x and 32GB RAM.

A. 108-Dimensional SRAM Column Circuit

An SRAM array is a typical type of random-access memory and uses flip-flops to store data. The SRAM array and cell are shown in Fig. 2, where WL is the word line, and BL is the bit line; two cross-connected inventors composed of four transistors

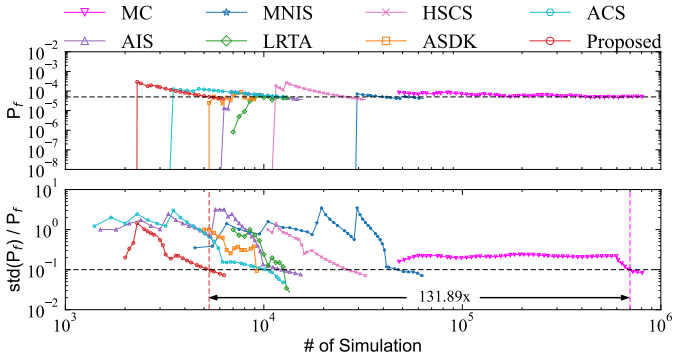

Fig. 3. P_f and FOM on 108-dimensional SRAM column

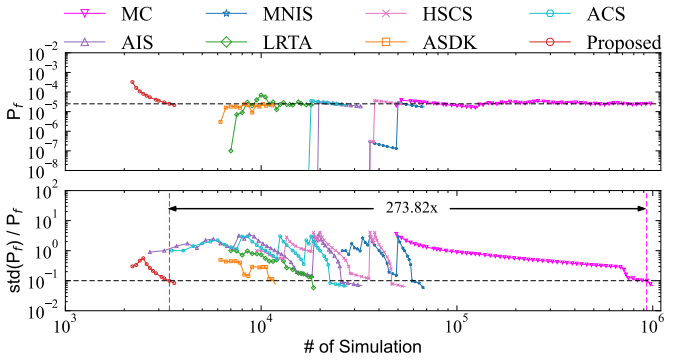

Fig. 4. P_f and FOM on 569-dimensional SRAM column

are used for storing data, whereas the other two transistors work as control switches for data transmission. Each transistor contains three variational parameters. An 8-bit SRAM array is composed of eight cells, resulting in 108 variation parameters. We choose the delay time of read/write of the SRAM as the output performance metric $\mathbf{y}$ of the circuit.

We show the failure probability estimation and FOM in Fig. 3, and the numerical results are concluded in Table I. The advantage of OPTIMIS is obvious by giving a 131.89x speedup compared to MC, more than twice faster than the second-best method AIS. Except for being the fastest, OPTIMIS also achieves the best accuracy among all methods with a relative error of 0.21%, more than 10x better than the second-best method HSCS. It is also interesting to see that OPTIMIS seriously overestimates the failure due to the suboptimal onion sampling. Such a bias is then sequentially reduced as more samples are collected.

B. 569-Dimensional SRAM Column Circuit

To validate OPTIMIS in practical scenarios, we work on a commercial SRAM array solution with 528 transistors in the design to form bit-cell arrays, sense amplifiers, and power paths. Based on the transistor type (PMOS/NMOS), gate length, and gate width, each transistor will associate with 0-3 variational parameters (i.e., mobility, oxide thickness, and saturation velocity) based on the BSIM model. With a BSIM4 model, this leads to 569 variation parameters. The delay time of read/write of the SRAM acts as the output metric $\mathbf{y}$. The convergence dynamic is shown in Fig. 4 with results in Table I. Similarly, the results of OPTIMIS are significantly better than the competitors, showing a 273.82x speedup over MC, which is almost 3.5x faster than the second-best method ASDK. Most importantly, the estimation results are very close to the ground truth, with a relative error of 0.25%. Again, the initial overestimation remains for OPTIMIS.

TABLE I
NUMERICAL RESULTS ON THREE SRAM COLUMN CIRCUITS

	108-dimensional case				569-dimensional case				1093-dimensional case			
Method	Fail. prob.	Rel. error	# of sim.	Speedup	Fail. prob.	Rel. error	# of sim.	Speedup	Fail. prob.	Rel. error	# of sim.	Speedup
MC	5.01e-5	-	699000	1x	2.50e-5	-	931000	1x	4.80e-5	-	1189000	1x
MNIS	4.15e-5	17.07%	47500	14.72x	2.07e-5	17.33%	59000	15.78x	4.21e-5	12.32%	81000	14.68x
HSCS	4.84e-5	3.36%	26500	26.38x	2.86e-5	14.27%	46500	20.02x	4.30e-5	10.47%	66000	18.02x
AIS	4.75e-5	5.21%	12300	56.83x	2.38e-5	4.99%	25700	36.23x	4.43e-5	7.75%	38000	31.29x
ACS	5.68e-5	13.40%	10400	67.21x	2.73e-5	9.19%	22500	41.38x	4.42e-5	7.83%	30400	39.11x
LRTA	4.50e-5	10.18%	13000	53.77x	2.26e-5	9.60%	18500	50.32x	5.25e-5	9.38%	24000	49.54x
ASDK	4.5e-5	10.18%	9200	75.98x	2.30e-5	8.00%	11800	78.90x	6.10e-5	27.08%	14550	81.72x
Proposed	5.02e-5	**0.21%**	**5300**	**131.89x**	2.49e-5	**0.25%**	**3400**	**273.82x**	4.67e-5	**2.71%**	**6400**	**185.78x**

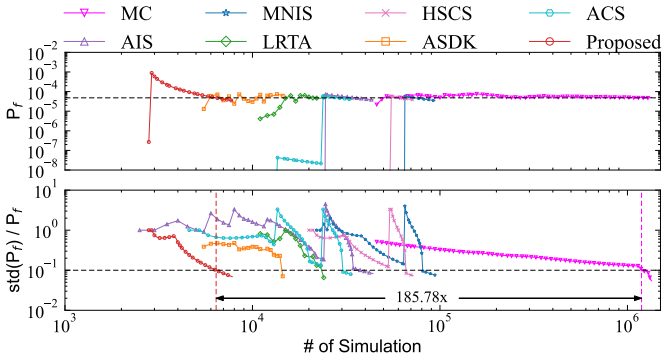

Fig. 5. P_f and FOM on 1093-dimensional SRAM column

TABLE II
ABLATION EXPERIMENT OF PRE-SAMPLING METHOD

	AIS	AIS+	Impro.	ACS	ACS+	Impro.
Rel. error	7.98%	6.79%	1.18x	6.59%	4.77%	1.38x
# of IS	13024	10854	1.20x	12100	9600	1.26x

C. 1093-Dimensional SRAM Column Circuit

We further increase the dimensions of the problem by using a detailed BSIM5 model, learning to 1093 variation parameters. As far as we are aware, no previous published work has ever attempted to estimate the yield of such a high-dimensional problem. Similarly, the delay time of read/write is used as the output metric. We run 1,189,000 simulations using the MC method and obtain the ground-truth failure rate, 4.80e-5. The results for the competing methods are shown in Fig. 5 and Table I. The superior OPTIMIS is consistent, although this time, the improvement over the second-best method ASDK is 2.2x, not as significant as in the previous experiments. Nevertheless, the ASDK shows the largest error of 27.08% among all methods.

D. Ablation Study

We validate the usefulness of the onion sampling by equipping the classic IS methods, namely, AIS and ACS ,as their pre-sampling procedure and compare with their original versions. The experiments are conducted on the same 108-dimensional SRAM experiment for its fast simulation speed. 1700 samples are given for all methods as their initial sampling budget, and the results are shown in Table II, which shows about 20% improvement in accuracy speedup with our onion sampling (AIS+ and ACS+).

E. Robustness Study

To further assess the robustness of our method, we conduct a robustness study on the 108-dimensional SRAM circuit by running experiments 10 times with random initializations. The final estimations with relative errors larger than 50% are marked fail. The statistical results for successful runs are shown in Table III. As we expect, the surrogate methods, LRTA and ASDK, suffer from great instability with more than 5 times fail, whereas the IS methods are more stable. In contrast, OPTIMIS shows the best performance for all metrics.

TABLE III
ROBUSTNESS TEST ON 108-DIMENSIONAL SRAM COLUMN

Method	MNIS	HSCS	AIS	ACS	LRTA	ASDK	Proposed
Avg. RE	16.00%	8.90%	6.64%	6.45%	12.85%	10.18%	**1.91%**
Avg. speedup	13.73x	22.30x	44.48x	49.68x	52.01x	75.98x	**131.89x**
# Fail	4/10	4/10	3/10	3/10	5/10	9/10	**1/10**

V. CONCLUSION

We generalize NM to optimal manifold and propose OPTIMIS to deliver the SOTA yield estimation. Limitation of OPTIMIS includes the implicit assumption of one failure boundary. Also, the NSF might need tuning for different problems, even though changing the NN structure in NSF does not have a significant influence on all the conducted experiments.

REFERENCES

[1] L. Dolecek, M. Qazi, D. Shah, and A. Chandrakasan, "Breaking the simulation barrier: Sram evaluation through norm minimization," in *2008 IEEE/ACM International Conference on Computer-Aided Design*. IEEE, 2008, pp. 322–329.

[2] W. Wu, S. Bodapati, and L. 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*, 2016, pp. 153–160.

[3] X. Shi, F. Liu, J. Yang, and L. 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)*. IEEE, 2018, pp. 1–6.

[4] X. Shi, H. Yan, J. Wang, X. Xu, F. Liu, L. Shi, and L. He, "Adaptive clustering and sampling for high-dimensional and multi-failure-region sram yield analysis," in *Proceedings of the 2019 International Symposium on Physical Design*, 2019, pp. 139–146.

[5] X. Shi, H. Yan, C. Li, J. Chen, L. Shi, and L. He, "A non-gaussian adaptive importance sampling method for high-dimensional and multi-failure-region yield analysis," in *2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, 2020, pp. 1–8.

[6] S. Yin, X. Jin, L. Shi, K. Wang, and W. W. Xing, "Efficient bayesian yield analysis and optimization with active learning," in *Proceedings of the 59th ACM/IEEE Design Automation Conference*, 2022, pp. 1195–1200.

[7] S. Yin, G. Dai, and W. W. Xing, "High-dimensional yield estimation using shrinkage deep features and maximization of integral entropy reduction," in *2023 28th Asia and South Pacific Design Automation Conference (ASP-DAC)*. IEEE, 2023.

[8] X. Shi, H. Yan, Q. Huang, J. Zhang, L. Shi, and L. He, "Meta-model based high-dimensional yield analysis using low-rank tensor approximation," in *Proceedings of the 56th Annual Design Automation Conference 2019*, 2019, pp. 1–6.

[9] J. Yao, Z. Ye, and Y. Wang, "An efficient sram yield analysis and optimization method with adaptive online surrogate modeling," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 7, pp. 1245–1253, 2015.

[10] C. Durkan, A. Bekasov, I. Murray, and G. Papamakarios, "Neural spline flows," *Advances in neural information processing systems*, vol. 32, 2019.