MINISTRY OF EDUCATION AND RESEARCH
NATIONAL UNIVERSITY OF SCIENCE AND TECHNOLOGY
BUCHAREST POLYTECHNIC

Doctoral School of Electronics, Telecommunications and Information Technology

Ph.D. THESIS
SUMMARY

Cristian Manolache

TEHNICI DE OPTIMIZARE ȘI ÎNVĂȚARE AUTOMATĂ APLICATE ÎN VERIFICAREA CIRCUITELOR INTEGRATE ANALOGICE

OPTIMIZATION TECHNIQUES AND MACHINE LEARNING APPLIED IN ANALOGUE CIRCUIT VERIFICATION

Prof. Dr. Ing. Corneliu Burileanu
Politehnica Univ. of Bucharest

PhD Supervisor

BUCHAREST 2025

Table of contents

1 Introduction **1**
- 1.1 Analog Circuits . . . 1
- 1.2 Machine Learning in Semiconductor Industry . . . 1
- 1.3 Motivation . . . 2
- 1.4 Objectives . . . 2
- 1.5 Thesis structure . . . 3

2 Computational Methods for Sampling and Optimization **4**
- 2.1 Sampling algorithms . . . 5
  - 2.1.1 Grid-Based Stratification (GBS) . . . 5
  - 2.1.2 Random Search . . . 5
  - 2.1.3 Latin Hypercube Sampling (LHS) . . . 5
  - 2.1.4 Orthogonal Array sampling . . . 5
- 2.2 Optimization algorithms . . . 6
  - 2.2.1 Gradient Descent . . . 6
  - 2.2.2 Evolutionary Algorithms . . . 6
- 2.3 Multi-Objective Acquisition Ensemble . . . 7
- 2.4 Gaussian Processes . . . 7
- 2.5 Circuit verification literature overview . . . 8

3 Testbench circuits **9**
- 3.1 General Overview . . . 9
- 3.2 Synthetic circuits . . . 9
- 3.3 Real circuits . . . 9
  - 3.3.1 Voltage Regulator . . . 10
  - 3.3.2 Feed-Forward Operational Amplifier (FFOA) . . . 10
- 3.4 Synthetic circuit benchmark . . . 10
  - 3.4.1 Introduction . . . 10
  - 3.4.2 Synthetic Benchmark Creation and Development . . . 10

4 Adaptive Verification for Analogue Circuits (AVAC) algorithm **11**
- 4.1 Fixed Planning . . . 11

4.2 Algorithm Adjustments 12
4.3 Adaptive Planning 13
4.4 Candidate selection using Multi-objective Acquisition Ensemble (MACE) 14
4.5 Process Corners 16

5 Algorithm Evaluation and Experimental Results 17
5.1 Evaluation metrics 17
5.2 Base version of the algorithm evaluation 17
5.3 Adjustments and Hyper-parameter Ablation Study 18
5.4 Adaptive Planning analysis 18
5.5 Candidate selection using MACE 18
5.5.1 Comparison between Gradient Descent and MACE approaches on the SC1-PC circuit 18
5.5.2 Comparison between Gradient Descent and MACE approaches on the VR circuits 19
5.5.3 Conclusions on MACE vs Gradient Descent approaches 19
5.6 Synthetic benchmark Validation 19
5.6.1 Analysis of Evaluation Sets of Small Dimensions 19
5.6.2 Overview of Higher-Complexity Evaluation Sets 19
5.7 Feed-Forward Operation Amplifier Verification 20
5.7.1 Initial Analysis 20
5.7.2 Verification Results for 4 Operating Conditions 20
5.7.3 Verification Results for 5 Operating Conditions 20

6 Risk estimation metric 21
6.1 Specification Failure Probability (SFP) 21
6.2 Experimental setup 21
6.3 Experimental Results 22
6.3.1 SC1 circuit with specifications that cannot be failed 22
6.3.2 SC1 circuit with specifications that can be failed 22
6.3.3 VR3 circuit with realistic specifications 23
6.4 Discussion 23

7 Simulation budget analysis 24
7.1 Adaptive Planning with Unequal Budget Allocation proposal 24
7.2 Proposed versions evolution on a simple scenario 24
7.3 Implemented dynamic budget version 4 24
7.4 Implemented version 4: evaluation and updates 25
7.5 Implemented dynamic budget version 5 25
7.6 Conclusions on dynamic budget allocation 25

8 Conclusions **26**
8.1 General Conclusions 26
8.2 Original contributions 26
8.3 List of original publications 27
8.4 List of contracts 28
8.5 Perspectives for further developments 29

References **30**

Chapter 1

Introduction

This thesis addresses the task of analog circuit verification in a Pre-Silicon environment with the aid of machine learning techniques. The research is motivated by the increasing complexity and strict reliability requirements of analog integrated circuits, particularly in the automotive industry.

1.1 Analog Circuits

Analog circuits are fundamental components in modern technology, essential for processing continuous signals in a vast range of applications. They are integral to automotive systems for engine control and safety features, audio equipment for high-quality sound amplification, industrial automation for precision control, and aerospace systems for navigation and communication. Their role in instrumentation, especially in medical and industrial devices, is also critical for accurate measurements and diagnostics.

1.2 Machine Learning in Semiconductor Industry

The semiconductor industry is increasingly leveraging machine learning to enhance manufacturing processes, improve production yields, and ensure product quality. Machine learning is applied in critical areas such as Design for Manufacturability (DFM), where models can automatically generate optimal layouts for analog circuits, replacing manual design [12]. It is also employed for yield prediction, where algorithms forecast production yields and analyze the impact of process parameters. Furthermore, machine learning techniques are used for defect and anomaly detection in wafer production, enabling early identification of process deviations to reduce costs and improve quality control.

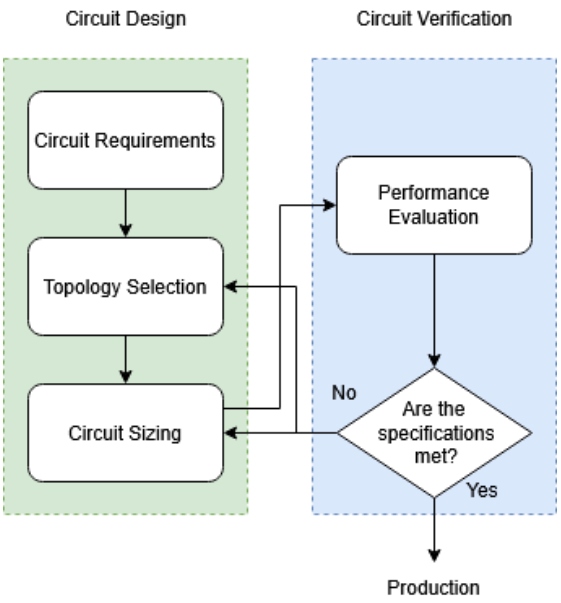


Fig. 1.1 Basic production pipeline of an analog circuit.

1.3 Motivation

The primary motivation for this work stems from the challenges in the automotive industry, where the addition of new features leads to greater design complexity and requires rigorous verification processes to ensure safety and reliability. Analog circuit verification is an important step in circuit development, ensuring that designs operate correctly according to their specifications. Figure 1.1 highlights the two main development steps of an analog circuit. Traditional verification methods rely on CAD tools and human input, often involving time-consuming grid-based explorations of the input space of operating conditions. While artificial intelligence has been applied to aid this process, an integrated and automated solution that can efficiently handle a large number of specifications and operating conditions is lacking. This thesis aims to adapt and apply artificial intelligence methods to create an automated tool that helps verification engineers find worst-case scenarios faster and more efficiently, a significant advantage in an industry with shrinking development times.

1.4 Objectives

The research presented in this thesis is guided by five primary objectives. The first objective is to research, design, and implement machine learning techniques for analog circuit verification, culminating in the development of the Adaptive Verification for Analog Circuits (AVAC) algorithm. This algorithm is intended to serve as an alternative to manual verification, achieving comparable or better results with fewer simulations by starting with an initial coverage of the input space. After simulating the circuit in

these initial points, the new results are used to further continue the search for worst-case corners. The second objective is to conduct an in-depth analysis of the AVAC algorithm, evaluating its baseline performance and exploring potential improvements through hyper-parameter tuning and other adjustments. The third objective involves the thorough evaluation and validation of the AVAC algorithm through the creation of a comprehensive synthetic benchmark of approximately 1000 functions that emulate real circuit responses of varying complexities. The fourth objective is to research, design, and implement a failure risk estimation metric, named Specification Failure Probability (SFP), to provide engineers with an early indication of the probability of specification failure, thus saving valuable time. The fifth and final objective is to create a simulation budget allocation strategy to optimize the efficiency of the AVAC algorithm, exploring how to best distribute a limited number of simulations among a certain circuit's various responses to identify potential failures more quickly.

1.5 Thesis structure

The thesis is organized into eight chapters. Chapter 2 introduces the theoretical notions of various sampling methods and optimization algorithms relevant to the circuit verification problem, alongside a review of existing literature. Chapter 3 presents the testbench circuits used for the evaluation of the AVAC algorithm, including both real and synthetic circuits, as well as a specially developed synthetic benchmark for exhaustive testing. Chapter 4 details the iterative development of the AVAC algorithm, from its initial version to subsequent updates and enhancements. Chapter 5 showcases the experimental results obtained by applying the different versions of the AVAC algorithm to the circuits described in Chapter 3. Chapters 6 and 7 introduce significant add-ons to the AVAC algorithm: Chapter 6 presents the SFP risk estimation metric, and Chapter 7 explores different strategies for dynamic simulation budget allocation. Finally, Chapter 8 presents the general conclusions of the research, outlines the original contributions, lists the publications related to the thesis, and discusses perspectives for future work.

Chapter 2

Computational Methods for Sampling and Optimization

Circuit verification aims to ensure that a circuit performs correctly according to its specifications across all possible input conditions, which are defined by variation variables like Process-Voltage-Temperature (PVT). Traditional verification methods, such as running a full factorial of all combinations, are thorough but often infeasible due to the exponential growth in simulations required with modern process nodes. Other approaches like a designer's "best guess" are fast but unreliable, while sensitivity analysis and quadratic models [21] may fail to capture the nonlinear behavior of circuits. This has led to the adoption of more complex regression models, such as Gaussian Processes [3], which can model arbitrarily nonlinear mappings from PVT variables to circuit outputs. The traditional design and verification flow, presented in Figure 2.1, is an iterative and time-consuming process, often taking months for even simple circuits. This thesis focuses on accelerating the verification phase, which occurs after the design parameters are fixed, by using machine learning to efficiently search the space of operating conditions (circuit inputs) for worst-case configurations that might lead to specification failures.

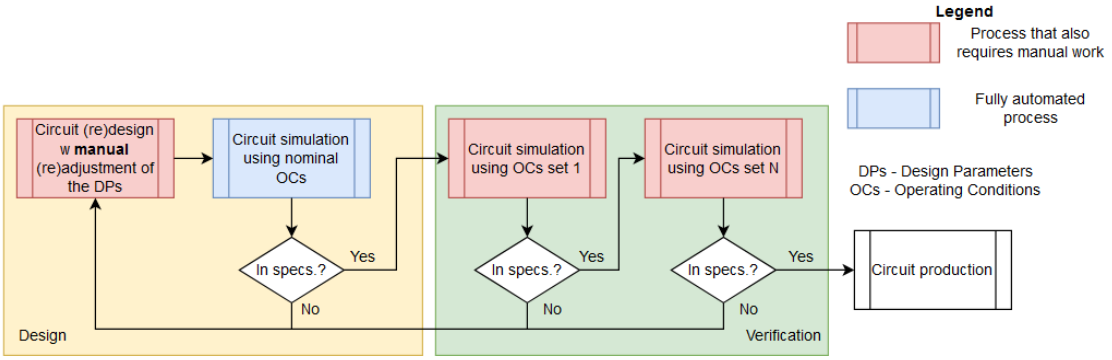


Fig. 2.1 Traditional design and verification flow for analog/mixed signal circuits

2.1 Sampling algorithms

To begin the verification process, an initial set of samples from the input space of operating conditions is required to gain insight into the circuit's behavior. The planning of these initial inputs is an important step for building an effective surrogate model [24]. Space-filling experimental designs, which aim to distribute sample points uniformly over the domain, are often preferred when little is known about the input-output relationship.

2.1.1 Grid-Based Stratification (GBS)

In GBS, each dimension of the input space is divided into equal parts, creating a grid of sub-cubes from which one point is sampled uniformly. This method offers a significant variance reduction compared to standard Monte Carlo sampling, especially in low dimensions. However, its practical utility is limited in high-dimensional spaces because the number of required samples grows exponentially (2^d, where d is the number of dimensions of the space we are sampling from), and it is difficult to extend an existing GBS sample set [23].

2.1.2 Random Search

Random search is a variation of GBS, that samples the search space randomly instead of using a discrete grid. While it also suffers from the curse of dimensionality, it is often a better practical choice than grid search because it explores distinct values for all parameters in each trial (attempt with a unique set of hyperparameters), making it more effective at optimizing functions with low effective dimensionality [2]. Its simplicity and that fact that it can be run in parallel make it an attractive first option, but it offers no guarantee of finding a local minimum without thorough sampling.

2.1.3 Latin Hypercube Sampling (LHS)

LHS is a stratification method where each of the d dimensions is divided into n equal strata, and n points are sampled such that each stratum in each dimension contains exactly one point. This method is particularly effective when the function to be modeled is nearly additive [23]. LHS can greatly reduce variance compared to plain Monte Carlo sampling without the risk of significantly increasing it, making it a suitable candidate for sampling in high-dimensional spaces.

2.1.4 Orthogonal Array sampling

Orthogonal Array sampling is a combinatorial method that generalizes LHS by stratifying all pairs (or t-tuples) of components. An orthogonal array of strength t ensures that in any sub-matrix of t columns, all possible combinations of levels appear an equal number

of times. This allows for the analysis of interactions between factors, not just their individual effects. Orthogonal arrays are particularly useful for detecting interesting behavior in extreme portions of the domain and have been shown to work well in intermediate and large dimensional spaces [23].

2.2 Optimization algorithms

After an initial sampling of the operating conditions hyperspace, a function optimization method is needed to drive the subsequent search for worst-case scenarios based on the initial data. Since circuit responses can be modeled as functions of the input operating conditions, these methods aim to find the minimum or maximum of an objective function.

2.2.1 Gradient Descent

Gradient descent is an optimization algorithm widely used in machine learning to minimize objective functions by iteratively adjusting parameters in the direction of the steepest descent of the function's gradient. There are three main types: Batch, Stochastic, and Mini-Batch Gradient Descent, which differ in how much data is used to compute the gradient for each parameter update. Variants like Momentum, RMSProp, and Adam have been developed to enhance its effectiveness. While simple and efficient, Gradient Descent is sensitive to the learning rate and can get stuck in local minima, but its modern variants have addressed many of these limitations, making it a powerful tool for the optimization problem.

2.2.2 Evolutionary Algorithms

Evolutionary algorithms are optimization methods inspired by natural selection, operating on a population of potential solutions that are progressively improved through processes like reproduction, mutation, and selection. They are particularly useful for complex optimization problems. Common operators include simulated binary crossover and polynomial mutation. The performance of these algorithms is influenced by several key parameters, such as crossover and mutation distribution indices, mutation probability, tournament size, and population size, which control the balance between exploration and exploitation.

Non-Dominated Sorting Genetic Algorithm-II (NSGAII) [4] is an evolutionary algorithm where each individual solution is assigned a rank based on the Pareto front it belongs to, with the best solutions having rank 0 (the Pareto front). Solutions that are only dominated by individuals from the Pareto front receive rank 1, and so on. The selection process for the next generation involves collecting all competing candidates and ranking them by Pareto dominance. If the number of non-dominated solutions

exceeds the population size, they are further ranked by distance and trimmed; otherwise, available places are filled with dominated solutions based on their quality.

Generalized Differential Evolution 3 (GDE3) [16] is an extended version of basic differential evolution, modifying its selection rule and applying it to problems with objective and constraint functions. In this algorithm, new candidate solutions are created by perturbing existing population members, typically using the weighted difference between two randomly selected individuals added to a third. The selection principle dictates that a trial vector replaces an old vector if it has an equal or lower objective value; if feasible solutions do not dominate each other, both offspring and parent vectors are saved for the next generation. This mechanism helps reduce computational costs and improve efficiency, and while the population size may increase, it is then reduced back to its original size by eliminating the worst members based on non-dominance and crowding. GDE3 has demonstrated improved solution diversity compared to NSGAII and better convergence properties in certain test problems.

2.3 Multi-Objective Acquisition Ensemble

Inspiration for an advanced candidate selection method was drawn from the 2020 NeurIPS Black-Box Optimization Challenge, where top-performing methods employed novel approaches to Bayesian optimization. One particularly influential method was the Multi-objective Acquisition Ensemble (MACE) algorithm Lyu et al. [19]. The MACE algorithm treats the optimization as a multi-objective problem, where three acquisition functions are combined to identify a Pareto front. The algorithm begins by creating a Gaussian Process model from an initial set of points and then constructs the three acquisition functions: Lower Confidence Bound (LCB), expected improvement, and probability of improvement. An evolutionary algorithm (DEMO [25]) is used to find the Pareto front based on these three functions, representing the optimal trade-offs between exploration and exploitation. A batch of points is then selected from this front for evaluation, and the process is iterated. This approach has demonstrated effectiveness on both synthetic benchmarks and real-world applications, proving to be robust and efficient in navigating high-dimensional and noisy objective spaces. The MACE algorithm is presented in Algorithm 1.

2.4 Gaussian Processes

To guide the optimization algorithms, a surrogate model is required to provide an estimate of the circuit response functions. Gaussian Processes were chosen for this role due to their roots in Bayesian inference, which allows them to provide not only predictions but also a reliable estimate of their own uncertainty [15]. This is a key benefit, as the uncertainty of a fitted GP increases in regions far from the training data. However, Gaussian Processes

Algorithm 1: Multi-objective Acquisition Ensemble Algorithm, where EI: Expected Improvement, PI: Probability of Improvement and GP: Gaussian Process.

Input: Number of initial sampling points N_{init}, number of iterations N_{iter}, batch size B

Output: best $f(x)$ recorded during iterations

```
Randomly sample N_init points in the design space
Construct initial GP model
for t = 1, 2, ..., N_iter do
    Construct the LCB, EI, and PI functions
    Find the Pareto front of LCB, EI, PI functions using the DEMO algorithm
    Randomly sample B points x_1, ..., x_B from the Pareto-optimal points
    Evaluate x_1, ..., x_B to get y_1 = f(x_1), ..., y_B = f(x_B)
    Update the GP model
end for
```

have drawbacks, including high computational cost in high-dimensional spaces and difficulty in handling categorical or integer-valued input variables. Naive approaches to handling such variables, like rounding, can lead to suboptimal results. More advanced methods modify the Gaussian Process kernel to ensure the model's behavior correctly reflects the discrete nature of the inputs [8].

2.5 Circuit verification literature overview

The task of circuit verification is critical, especially in safety-critical domains, and can consume up to 70% of the total design time [5] [7]. The circuit verification process is split into 2 sub-stages, namely, the Pre-Silicon verification stage and the Post-Silicon verification stage, with our focus on the former. While verification is applied to both digital [14] [9] and analog circuits, the latter are more complex [10] due to their continuous output values. Advanced approaches for analog verification often employ machine learning techniques [13] [6] [32] [26]. For instance, some methods use sequential pattern mining to maximize test coverage [18], while others propose Hybrid Formal/Machine-learning Verification (HFMV) frameworks [13] that use probabilistic models like Gaussian Processes to find failures more efficiently than Monte Carlo methods. The adjacent domain of circuit sizing also benefits from similar techniques, with evolutionary algorithms and Bayesian optimization being used to find optimal design parameters under a limited simulation budget [27] [29].

Chapter 3

Testbench circuits

3.1 General Overview

In the context of Pre-Silicon verification, a circuit is defined by three main characteristics: design parameters, operating conditions, and responses. Design parameters have fixed values set during the design phase and do not vary during verification. Operating conditions are the continuous or categorical input variables (such as temperature or process corners) that are varied to test the circuit's robustness. Responses represent the circuit's outputs, which must remain within certain specification thresholds for the circuit to pass verification. The algorithm's performance is evaluated using two categories of circuits: synthetic circuits, where the responses are known mathematical functions of the inputs, and real circuits, which are models simulated in CAD tools.

3.2 Synthetic circuits

Synthetic circuits are highly advantageous for algorithm evaluation because their responses are known functions of input operating conditions, allowing for the precise calculation of true minimum and maximum values. This enables an accurate assessment of how close the algorithm gets to the true worst-case values of the responses. A key synthetic circuit used in the experiments is Synthetic Circuit 1 (SC1), which has six operating conditions and 25 responses defined as linear combinations of functions dependent on a single operating condition. A variant, SC1-PC, includes an additional process corner to emulate manufacturing variations.

3.3 Real circuits

To validate the algorithm's applicability in real-world scenarios, two types of real circuits were used. These "real" circuits are models obtained after the design phase and are

simulated in CAD tools. Unlike synthetic circuits, the true minimum and maximum values of the responses for these real circuits are generally unknown.

3.3.1 Voltage Regulator

The first real circuit is a Voltage Regulator (VR), for which three different variants (VR1, VR2, VR3) are used. Each VR circuit has 27 design parameters, seven operating conditions, one process corner, and three responses with specification limits that must be respected: gain margin $>$ 8 dB, phase margin $>$ 20 deg., and Power Supply Rejection Ratio (PSRR) $<$ –26 dB.

3.3.2 Feed-Forward Operational Amplifier (FFOA)

The second real circuit is a feed-forward operational amplifier (FFOA), which is an open-source design [28]. The FFOA is defined by 11 fixed design parameters, a variable number of operating conditions (between two and five), a process corner, and six responses, including current consumption (IDD), unity-gain bandwidth (f0dB), DC gain (LG0), Phase Margin (PM), Gain Margin (GM), and Slew Rate (SR).

3.4 Synthetic circuit benchmark

3.4.1 Introduction

While synthetic benchmarks for digital circuits are available in the literature, they are rare for Pre-Silicon analog circuit verification. To address this gap, a comprehensive benchmark was designed and implemented in collaboration with industry engineers to rigorously test the AVAC algorithm.

3.4.2 Synthetic Benchmark Creation and Development

A data-driven approach was used to create a benchmark of 900 synthetic circuits, with 100 circuits for each number of input operating conditions, ranging from two to ten. The benchmark was built from a base list of 30 one-dimensional functions designed by circuit engineers to resemble plausible real circuit behaviors. The complexity of these base functions was assessed through a subjective grading process by a group of six people. The final synthetic circuits (one response) were created by adding these one-dimensional functions together. To compare the difficulty of circuits with different numbers of inputs, a metric called Multiplicative Complexity (MCpx) was defined as the product of the complexities of the component functions. The benchmark was designed to focus on circuits with a maximum MCpx of 40 and up to 10 operating conditions, reflecting realistic scenarios.

Chapter 4

Adaptive Verification for Analogue Circuits (AVAC) algorithm

This chapter details the Adaptive Verification for Analogue Circuits (AVAC) algorithm, whose main objective is to verify that a circuit's responses remain within specifications across all operating conditions. The algorithm treats circuits as black boxes and follows a three-step process: (1) Fixed Planning for initial data acquisition, (2) training surrogate models, and (3) applying a candidate proposal algorithm to find potential specification failures. The main steps of the verification method are also highlighted in Figure 4.1.

4.1 Fixed Planning

The Fixed Planning stage serves as the initial phase of the AVAC algorithm, focusing on generating an initial dataset for training surrogate models. Traditionally, verification engineers relied on a full-factorial approach, simulating all combinations of minimum, maximum, and nominal values for operating conditions. However, when the number of operating conditions is large, this method is not feasible because the number of simula-

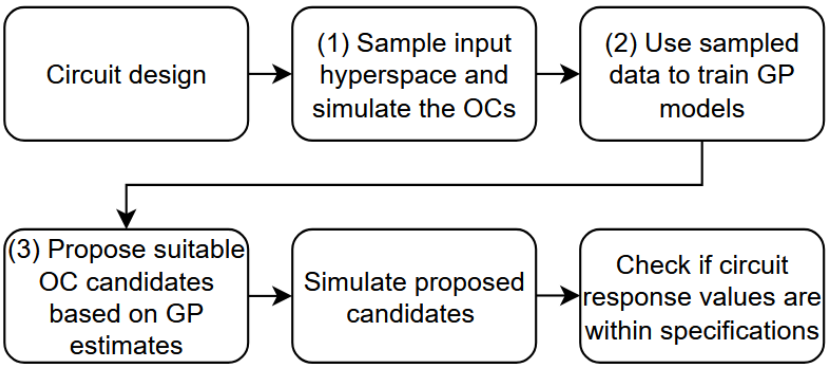


Fig. 4.1 Main steps of the baseline AVAC algorithm. Where OC: Operating Condition; GP: Gaussian Process.

tions grows exponentially. To overcome this, alternative sampling methods are employed, including Monte Carlo, Orthogonal Array sampling [20], and Latin Hypercube Sampling (LHS) [22]. Orthogonal array sampling is particularly effective for exploring corners of the operating condition hyperspace with fewer simulations, while LHS provides better overall coverage by stratifying each dimension. Another approach considered consists in combining orthogonal array and LHS to leverage their respective strengths, aiming for both corner exploration and comprehensive hyperspace coverage.

The data collected during the Fixed Planning stage is then used to train Gaussian Process surrogate models for each circuit response. Gaussian Processes are favored due to their ability to model functions probabilistically and provide a reliable estimate of their own uncertainty, which increases further from the training data. To guide the search for potential circuit failures, an acquisition function called the Lower Confidence Bound (LCB) is utilized. The LCB function, estimates the lower envelope of the Gaussian Process uncertainty. The beta parameter for the LCB allows for adjusting the balance between exploiting known low-value regions (smaller beta) and exploring uncertain areas (larger beta). The BoTorch framework [1] is used for the implementation of these Gaussian Process models and acquisition functions.

Following the training of the Gaussian Process models, a candidate proposal algorithm is applied to suggest Operating Condition Configurations (OCCs) that are likely to lead to circuit specification failures. This process begins by generating a large evaluation set of OCCs, typically a combination of a full-factorial grid and additional LHS samples. The trained Gaussian Processes then predict circuit responses and their associated uncertainties for these candidate OCCs. These candidates are ranked using the LCB score, and the OCC with the lowest LCB score (for minimization problems) is selected as the next candidate for actual circuit simulation.

4.2 Algorithm Adjustments

Several adjustments to the baseline algorithm were considered to improve its performance. One key update was to apply gradient descent to the initial set of evaluation candidates. This method, typically employed for function minimization, leverages Gaussian Process estimates to direct the search toward operating conditions that yield response values closer to the function's minimum, thereby refining the pool of potential candidate solutions.

Further updates included an ablation study on critical hyper-parameters within the updated candidate proposal algorithm. This investigation encompassed the beta parameter for the LCB acquisition function, the learning rate, the choice of optimizer (Adam or Stochastic Gradient Descent), and the number of gradient descent steps. While Adam is recognized for its computational speed, SGD is noted for its superior generalization capabilities [11] [31]. Additionally, alternative acquisition functions beyond LCB were

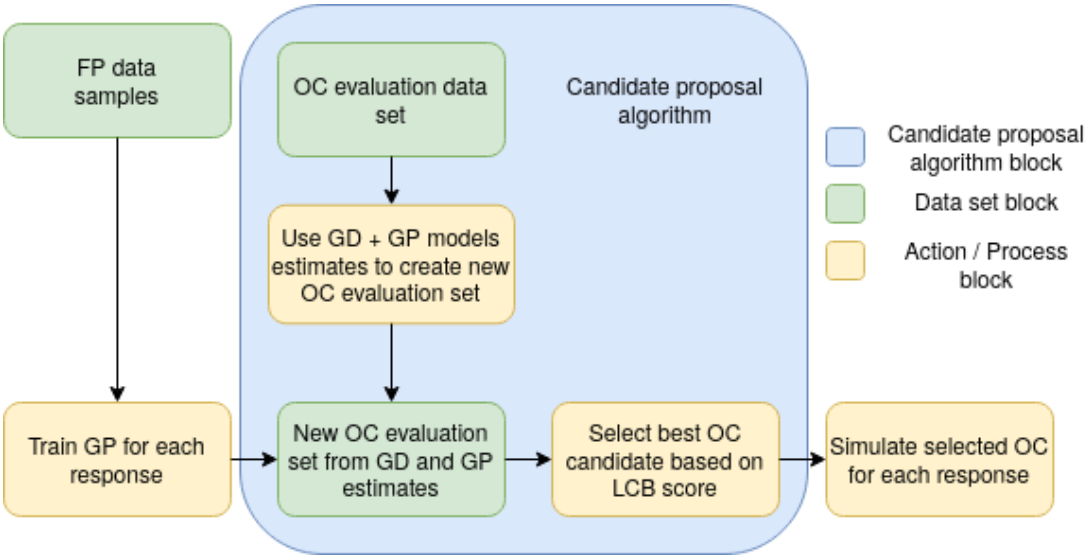


Fig. 4.3 Candidate selection algorithm diagram, where FP: Fixed Planning, OC: Operating Condition, GD: Gradient Descent, GP: Gaussian Process.

considered, for ranking operating conditions, specifically probability of improvement and expected improvement. The probability of improvement represents the probability of surpassing the best observed value, while the expected improvement measures the expected magnitude of such improvement. An overview of the full algorithm, with the updated candidate selection method is presented in figure 4.3.

4.3 Adaptive Planning

This section introduces a new stage, called Adaptive Planning (Figure 4.4), as an iterative phase designed to overcome a limitation in earlier algorithm versions, where Gaussian Process models performed only a single OCC selection per response. Initially, GP models were trained using Fixed Planning data, derived from a combination of orthogonal array sampling and LHS. Subsequently, gradient descent was applied to an evaluation set of candidate OCCs to identify those most likely to induce circuit failures.

The core mechanism of Adaptive Planning involves to iteratively retrain the Gaussian Processes over multiple cycles. In each iteration, the models are updated by incorporating both the initial Fixed Planning samples and all new candidate OCCs simulated in previous iterations. This continuous refinement allows the Gaussian Processes to progressively enhance their predictive accuracy for circuit responses, leading to the proposal of better candidates that may lead to worst-case scenarios. The number of Adaptive Planning iterations is a configurable parameter, determined by a human expert overseeing the verification process.

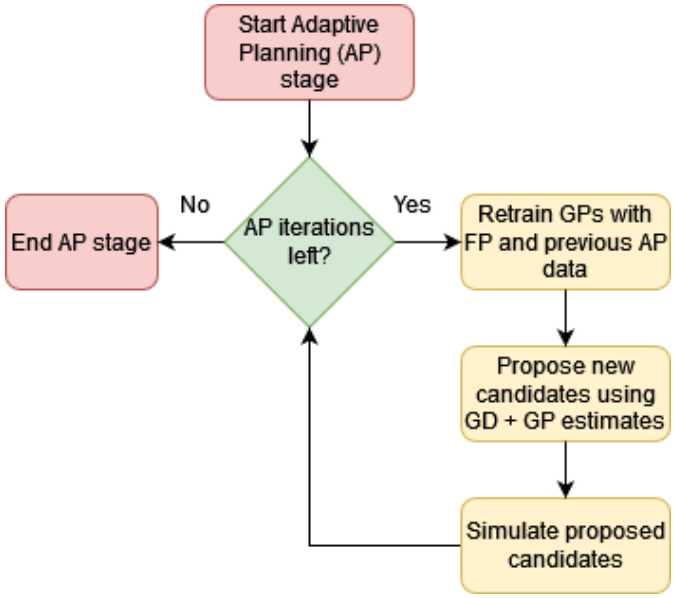


Fig. 4.4 Adaptive planning stage diagram. The number of Adaptive Planning iterations is chosen by the human expert which conducts the verification process. FP: Fixed Planning, GD: Gradient Descent, GP: Gaussian Process.

4.4 Candidate selection using Multi-objective Acquisition Ensemble (MACE)

The baseline algorithm's candidate selection process, which utilized gradient descent and a single acquisition function (such as LCB, probability of improvement, or expected improvement), presented several drawbacks, including potential convergence to local minima and a lack of robustness across different scenarios. To address these limitations, a modified version of Multi-objective Acquisition Ensemble (MACE) algorithm [19] [30] was introduced. This approach, considers the candidate selection as a multi-objective optimization problem. It aims to combine the strengths of multiple acquisition functions to achieve a more robust and effective search for circuit failures.

The MACE variant of the algorithm starts by creating an initial population of OCCs using a combination of grid-based and LHS. These candidates, along with previously simulated points, are then evaluated using three distinct acquisition functions: LCB, expected improvement, and probability of improvement. The algorithm identifies the Pareto front from these multi-objective evaluations using the GDE3 algorithm [17], balancing exploration of uncertain regions with exploitation of promising ones. A batch of points is then selected from this Pareto-optimal set for actual circuit simulation, iteratively updating the Gaussian Process models to refine the search. The main points of the alternate MACE variant used in the AVAC algorithm are presented in Algorithm 2, while a diagram of the full algorithm is provided in in Figure 4.5.

Algorithm 2: Alternate MACE variant, where OC: Operating Condition and GP: Gaussian Process.

Input: Initial population of size N, trimmed population of size P, number of GDE3 iterations I

Output: Best B OC candidate solutions

1 An initial population of OC candidates is created. Starting from a L level grid per input dimension, the remaining points are filled up to N using LHS.
2 Add simulated points from previous Adaptive Planning iterations.
3 Evaluate the starting population using the GP for all three (expected improvement, probability of improvement, LCB) acquisition functions.
4 Trim the starting population to the best P solutions using non-dominated pruning.
5 Use GDE3 to evolve the P solutions for I iterations/epochs.
6 The I populations from each epoch are combined and filtered for duplicates resulting in a final OC candidate set.
7 Select the best B solutions from the final OC candidate set.

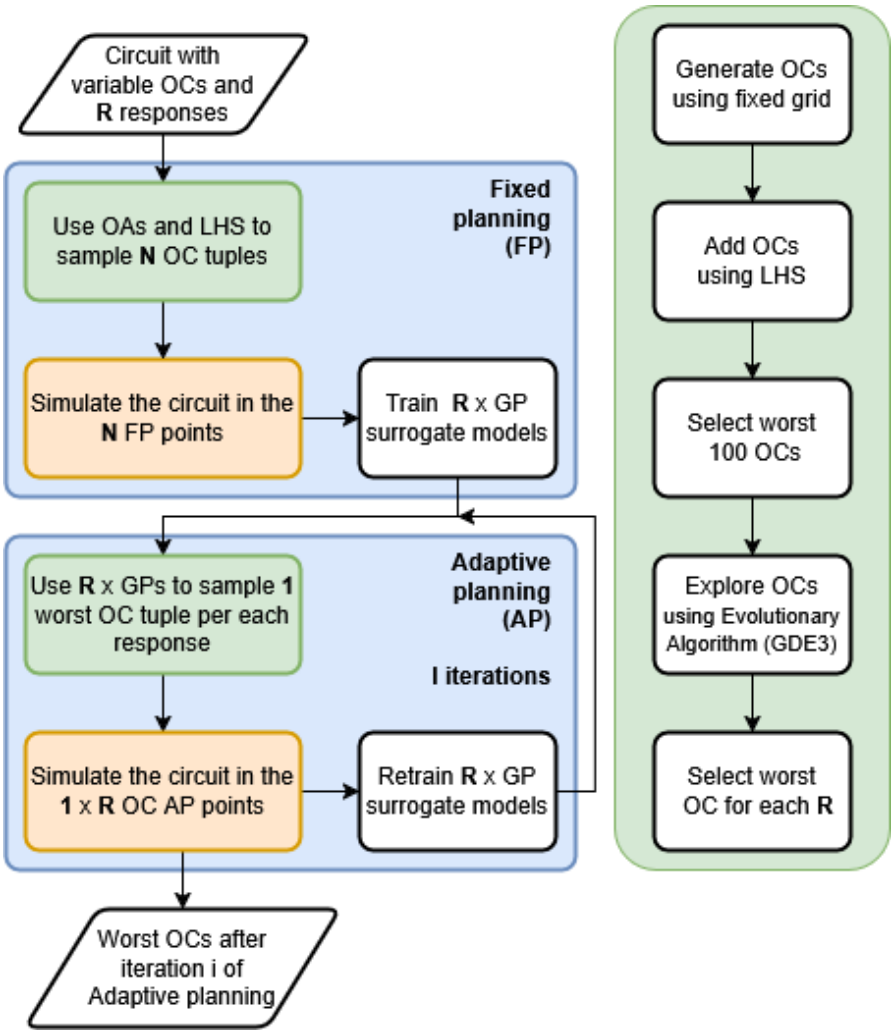


Fig. 4.5 Diagram of the updated candidate selection algorithm, with MACE block (green) on the right. FP: Fixed Planning, OA: Orthogonal Array, OC: Operating Condition, GP: Gaussian Process.

4.5 Process Corners

The algorithm was extended to handle process corners, which are categorical variables representing manufacturing variations that can significantly impact circuit performance. Instead of treating them as unordered categories, a more efficient encoding scheme was developed that represents the corners in a multi-dimensional integer space. This approach takes advantage of the inherent ordering between corner values (e.g., slow, typical, fast) and allows the Gaussian Process model to learn the relationships between them more effectively. Mechanisms were also implemented to handle the sparse nature of the valid corner grid, such as assigning poor scores to invalid combinations generated during the evolutionary optimization process.

Chapter 5

Algorithm Evaluation and Experimental Results

This chapter presents the experimental results and evaluation of the Adaptive Verification for Analogue Circuits (AVAC) algorithm throughout its various development stages. It outlines the algorithm's performance on both synthetic and real circuits (described in Chapter 3), starting with the base version and progressing through updates involving the gradient descent and the Multi-objective Acquisition Ensemble (MACE) variants. The chapter also introduces the specific evaluation metrics used to assert the algorithm's effectiveness.

5.1 Evaluation metrics

The performance of the algorithm was assessed using different metrics for synthetic and real circuits. For synthetic circuits, where the true minimum and maximum response values are known, the primary metric was the Relative Value Error (RVE), which measures the normalized difference between the value found by the algorithm and the true minimum. Two aggregate metrics, Average RVE (ARVE) and Maximum RVE (MRVE), were also used to evaluate performance across all responses of a circuit. The ARVE represents the average RVE across all of a circuit's responses, while MRVE represents the maximum RVE. For real circuits, where true extrema are unknown, the evaluation was based on the actual worst-case values found and whether they failed the imposed specifications.

5.2 Base version of the algorithm evaluation

Initial experiments on the SC1 synthetic circuit compared different sampling methods for the Fixed Planning stage. The key finding was that a combination of orthogonal array and LHS sampling provided the best initial coverage and outperformed full-factorial and

Monte Carlo approaches, especially with low simulation budgets. Subsequently, training Gaussian Process models on this data and proposing a single new candidate for each response resulted in a significant improvement, achieving an ARVE of under 1% with far fewer simulations than the classical full-factorial approach.

5.3 Adjustments and Hyper-parameter Ablation Study

Further experiments focused on tuning the algorithm have shown that varying the beta hyper-parameter for the LCB acquisition function had a relatively minor impact on performance. The most significant improvement came from applying gradient descent to the evaluation set of Operating Condition Configurations (OCCs) to generate better candidates. An ablation study on the gradient descent optimizers (Adam and SGD) and various learning rates revealed that specific configurations could further reduce the ARVE, with the best results reaching 0.13% ARVE and 0.62% MRVE.

5.4 Adaptive Planning analysis

The introduction of the full Adaptive Planning stage, with multiple retraining iterations, was evaluated on the SC1-PC synthetic circuit and the VR2 real circuit. On the synthetic circuit, running for 10 iterations reduced the ARVE from 0.75% after one iteration to 0.35%. On the VR2 circuit, the multi-iteration process was crucial for finding a specification failure for the gain margin response, a failure that was missed with only one iteration. This demonstrated that iteratively refining the surrogate models with new data can lead to the uncovering of subtle failures in complex circuits.

5.5 Candidate selection using MACE

This section presents experimental results comparing the baseline (gradient descent) and updated (MACE) versions of the AVAC algorithm on both the SC1-PC synthetic circuit and all three variants of the VR real circuits. All experiments used the same setup, with only the candidate selection approach and the number of iterations being varied.

5.5.1 Comparison between Gradient Descent and MACE approaches on the SC1-PC circuit

On the SC1-PC synthetic circuit, the MACE approach demonstrated slightly better performance. After one iteration of Adaptive Planning, MACE achieved a better ARVE (0.38% vs. 0.71%) than the gradient descent variant. Over 10 iterations, MACE reached its best result of 0.12% ARVE (vs 0.31%) faster (at iteration 7) and found the true

minima for more of the circuit's responses (18 out of 25) compared to the gradient descent approach (12 out of 25).

5.5.2 Comparison between Gradient Descent and MACE approaches on the VR circuits

On the three variants of the VR real circuit, the results were more nuanced, with both methods performing comparably on several responses. However, a key finding emerged from the tests on the VR1 circuit: after 10 iterations, the MACE approach successfully identified an OCC that led to a specification failure for the gain margin response, while the gradient descent approach did not find this critical failure.

5.5.3 Conclusions on MACE vs Gradient Descent approaches

While the MACE algorithm is more complex, its multi-objective nature proved to be slightly more robust, particularly in its ability to identify critical specification failures that were missed by the single-objective gradient descent method. It is hoped that MACE's may perform better in verifying more complex circuits, as the tested circuits were relatively simple, where the gradient descent approach was "good enough".

5.6 Synthetic benchmark Validation

This section validates the algorithm's performance across all nine evaluation sets of the synthetic benchmark, which include 900 synthetic circuits with varying complexities and 2 to 10 input operating conditions. All experiments utilized the latest MACE version of the verification algorithm, with each function undergoing 100 Fixed Planning and 100 Adaptive Planning simulations.

5.6.1 Analysis of Evaluation Sets of Small Dimensions

Initial tests on circuits with two and three operating conditions showed that the algorithm performed well, with the number of iterations required to find a near true minimum value (RVE < 2%) correlating with the subjective complexity of the underlying functions.

5.6.2 Overview of Higher-Complexity Evaluation Sets

Next, the algorithm was evaluated on more complex sets of circuits (4 up to 10 operating conditions). For the majority of circuits (over 70% for sets up to seven operating conditions), it successfully found a near true minimum value in under 10 Adaptive Planning iterations. As the number of operating conditions and the complexity of the circuits increased, more iterations were required, which is expected due to the

higher dimensionality of the search space. Overall, the successful validation on this diverse benchmark confirmed the algorithm's capability and consistency for Pre-Silicon verification tasks.

5.7 Feed-Forward Operation Amplifier Verification

This section details the verification process for the FFOA real circuit, employing the latest MACE version of the algorithm to find results comparable to or better than a classical approach (full-factorial) but with fewer simulations. The goal is to identify similar or improved outcomes compared to classical verification while using a reduced number of simulations, with varying configurations for both approaches.

5.7.1 Initial Analysis

An initial analysis with two operating conditions and five process corners found no specification failures, although the phase margin response was observed to be very close to its limit. A more detailed grid search confirmed that the worst-case value was indeed a corner and did not fail the specification.

5.7.2 Verification Results for 4 Operating Conditions

The verification was expanded to include four operating conditions by treating two previously fixed parameters as variable. In this more complex scenario, the AVAC algorithm successfully identified a specification failure for the phase margin response. The algorithm found this failure using 160 simulations, which is less than 35% of the 486 simulations required by a classical full-factorial approach. It also identified a failure at a non-corner OCC, a point that a standard corner-based verification would have missed.

5.7.3 Verification Results for 5 Operating Conditions

Finally, a fifth operating condition was added, further increasing the problem's complexity. The algorithm obtained similar results to the exhaustive full-factorial search, this time finding failures for both the phase margin and the slew rate responses. This was achieved with 160 simulations, approximately 10% of the 1458 simulations needed for the full-factorial baseline, highlighting the algorithm's efficiency.

Chapter 6

Risk estimation metric

This chapter introduces a novel metric, Specification Failure Probability (SFP), which indicates the probability of a circuit failure during the Pre-Silicon verification process. The metric is designed to be used with any verification algorithm that employs Gaussian Processes, leveraging their ability to provide uncertainty estimates alongside their predictions.

6.1 Specification Failure Probability (SFP)

The SFP is defined as the probability that a circuit's response will not meet its specification threshold. It is computed for a given Operating Condition Configuration (OCC) by using the Gaussian Process model's predicted mean and variance to determine the probability of the response value falling into the failure region. To provide a single, comprehensive risk estimate for an entire response, a total SFP is calculated. This is done by sampling a large set of OCCs, clustering them to avoid bias from correlated points, identifying the worst-case SFP in each cluster, and then combining these individual probabilities to obtain a total SFP for the response. An overview on how the Total SFP is calculated, is presented in Algorithm 3.

6.2 Experimental setup

The experimental validation of the SFP metric focused on two distinct circuit types: synthetic and real circuits, chosen to provide both controlled and realistic assessment contexts. The Synthetic Circuit 1 (SC1) was employed first due to its rapid function evaluation time and the known true worst-case values. This characteristic allowed for a clear and reliable evaluation of the SFP metric's accuracy. Complementing this, a Voltage Regulator (VR3), representing a model of a real circuit, was utilized, featuring three key responses: gain margin, phase margin, and Power Supply Rejection Ratio (PSRR). These responses were subject to specific operational thresholds, such as gain margin > 8

Algorithm 3: Specification Failure Probability

Input: A set of N OCCs data points comprised from $3^L + previous\ Adaptive\ Planning\ OCCs$ candidates, where L is the number of input operating conditions

Output: Total SFP value for a response

1 Calculate the SFP value for each data point as the probability of the estimate to exceed the specification.
2 Create 10 clusters using K-means, based on the data points position in the hyperspace.
3 Create the SFP vector of length 10 based on the K-means clusters.
4 **for** $i = 1$ **to** *10* **do**
5 | Add the data point with the best SFP value to the SFP vector
6 **end for**
7 Calculate the Specification Non-Failure Probability (SNFP) as $SNFP = 1 - SFP$
8 Calculate the Total SFP value for the response as: $Total\ SFP = 1 - \prod(SNFP)$

dB, phase margin > 20 deg., and PSRR < –26 dB. In both scenarios, the circuits were treated as "black boxes," with their inputs confined within defined limits. The reliability of the SFP was evaluated by comparing its predicted values against known or estimated response limits, including those obtained during Fixed Planning and Adaptive Planning iterations, as well as the true minimum or maximum values when available. A higher SFP was expected as response values approached or failed the specification thresholds.

6.3 Experimental Results

6.3.1 SC1 circuit with specifications that cannot be failed

The experimental evaluation of the SFP metric started with the SC1 synthetic circuit, configured such that the specifications of its 25 responses were deliberately set to 2% (of the respective range of the response) below their true minima, rendering them not failable. Initial results showed that several responses exhibited SFP values exceeding 10% after the Fixed Planning stage and in early Adaptive Planning iterations. By AP iteration 4, all responses reached SFP values below 1.51%, indicating a minimal probability of specification failure, consistent with the circuit's design. This shows the SFP's utility in signaling when further verification efforts would be unnecessary.

6.3.2 SC1 circuit with specifications that can be failed

In a subsequent experiment, the SC1 circuit was configured with specifications set at the true minimum plus 1% of the response range, making failures possible. The expectation was to observe higher SFP values as the algorithm approached these specifications. Due to the relative simplicity of some responses, the verification algorithm identified failures

for most responses during the Fixed Planing stage. For the remaining responses where failures were not found in Fixed Planning, the SFP metric accurately presented high values, justifying continued simulation. The algorithm successfully identified OCCs leading to specification failures for all these responses by Adaptive Planning iteration 1. It was noted that for more complex responses, initially lower SFP values indicated the need for additional Adaptive Planning iterations to sufficiently train the Gaussian Process models for reliable decision-making.

6.3.3 VR3 circuit with realistic specifications

Finally, the SFP metric was applied to a real circuit, the VR3 Voltage Regulator, using realistic specifications for its three responses: gain margin, phase margin, and PSRR. For the gain margin response, consistently high SFP values, starting at 97.5% during Fixed Planning and remaining high, accurately predicted a high probability of failure. This was validated at Adaptive Planning iteration 48 when an OCC was discovered that resulted in a gain margin value of 7.92 dB, violating the >8 dB specification. Conversely, the phase margin response's SFP, initially at 64.7%, steadily decreased over 50 iterations, correctly indicating no specification failures were found (28.6 deg vs. >20 deg specification). Similarly, the PSRR response's SFP remained low, dropping to 0% by iteration 20, aligning with the actual response value of -34.1 dB, which was well within the <-26 dB specification. These results collectively show that the SFP can guide the verification engineers to effectively assess the probability of a circuit failure.

6.4 Discussion

The experimental results showed that the SFP metric is a reliable and objective tool that can provide valuable insight to the verification engineer. It successfully predicts the outcome in various scenarios, issuing timely warnings of potential failures or indicating when a circuit is robust enough to stop the verification process early. While the absolute value of SFP is dependent on the quality of the surrogate model, it serves as a good indicator for comparing failure likelihood and making informed decisions during the verification process.

Chapter 7

Simulation budget analysis

7.1 Adaptive Planning with Unequal Budget Allocation proposal

This chapter addresses the challenge of optimally allocating a limited simulation budget during the Adaptive Planning stage of the Adaptive Verification for Analogue Circuits (AVAC) algorithm. The standard approach allocates one simulation to each response in every Adaptive Planning iteration. This can be inefficient, especially if some responses have already failed their specification or have a very low probability of failure. The central idea proposed in this chapter is to dynamically re-allocate simulations from responses that have already failed or are considered low-risk to those that have a high Specification Failure Probability (SFP), with the goal of finding potential failures more quickly.

7.2 Proposed versions evolution on a simple scenario

Several strategies for dynamic budget allocation were considered and analyzed. The proposed versions evolved from a simple approach of removing failed responses from the simulation budget (reducing overall time or allowing for extra iterations) to a more complex strategy of actively re-allocating free simulation slots to the remaining valid responses (respective specification not failed). The most advanced of these initial proposals involved using the SFP value of each valid response to guide the reallocation, giving more simulations to responses with a higher perceived risk of failure.

7.3 Implemented dynamic budget version 4

Based on the analysis of the proposed versions, a comprehensive dynamic budget allocation strategy, referred to as version 4, was implemented. This version operates in

several steps. First, any response that has failed its specification receives zero simulations for all future iterations. The remaining valid responses each receive a baseline of one simulation. The extra simulations remaining from the failed responses are then distributed among the valid responses in a weighted manner based on their SFP values from the previous iteration, with responses having a higher SFP receiving more simulations. Any remaining simulations after this step are allocated randomly.

7.4 Implemented version 4: evaluation and updates

The experimental evaluation of version 4 demonstrated its effectiveness. Compared to the standard budget allocation where each response gets one simulation per iteration, the unequal allocation strategy found all specification failures in only four iterations, whereas the standard method required seven. This confirmed that strategically allocating the budget based on SFP can significantly accelerate the discovery of failures. A further update was implemented to allow SFP-based allocation to be used from the very first iteration, which improved performance even more, finding all failures in just three iterations.

7.5 Implemented dynamic budget version 5

A shortcoming of version 4 was that it always allocated at least one simulation to every valid response, which could be wasteful for responses with a near-zero chance of failure. To address this, version 5 was developed. This version refines the logic for scenarios where the number of available parallel simulations is smaller than the number of valid responses. In such cases, version 5 allocates simulations only to the responses with the highest SFP values, completely omitting those with low SFP and avoiding the creation of extra simulation batches.

7.6 Conclusions on dynamic budget allocation

The research into dynamic budget allocation demonstrates that moving away from a static, equal allocation of simulations is highly beneficial. The proposed methods, particularly versions 4 and 5, offer robust alternatives that leverage the SFP risk metric to prioritize simulations intelligently. By focusing the simulation budget on high-risk responses, these strategies can considerably reduce the time and number of simulations required to identify potential specification failures, making the overall verification process more efficient.

Chapter 8

Conclusions

8.1 General Conclusions

The main objective of this thesis consists in the development of a machine learning-based algorithm for the Pre-Silicon verification of analogue integrated circuits. The research was motivated by the increasing complexity and strict reliability demands of the automotive industry. The developed Adaptive Verification for Analogue Circuits (AVAC) algorithm was built through an iterative process, starting from a baseline version and evolving with several enhancements. Key steps included the evaluation of various sampling strategies for initial data collection, with a combination of Orthogonal Arrays and Latin Hypercube Sampling proving most effective. The algorithm was then improved by introducing an Adaptive Planning stage, which allows the surrogate models to be retrained iteratively. Next, an advanced candidate selection mechanism based on the Multi-Objective Acquisition Ensemble (MACE) approach, which proved slightly better than the single-objective gradient descent method. The algorithm's performance was validated on both real circuit models and a comprehensive synthetic benchmark of 900 functions, demonstrating its robustness and efficiency. To further enhance the verification process, two important extensions were developed: the Specification Failure Probability (SFP) metric, which provides an estimate of the failure risk, and a dynamic simulation budget allocation strategy that uses the SFP to prioritize simulations and find failures faster.

8.2 Original contributions

The personal contributions of the author of this thesis can be summarized as follows:

1. Developed a Machine Learning based adaptive verification procedure for analogue circuits.

(a) Implemented and evaluated various strategies for covering the Operating Conditions hyperspace during the Fixed Planning stage of the Adaptive Verification for Analogue Circuits algorithm, such as Orthogonal Arrays, Latin Hypercube Sampling and Monte Carlo.

(b) Implemented and evaluated the baseline version of the Adaptive Verification for Analogue Circuits algorithm based on Gradient Descent.

(c) Implemented and evaluated the MACE version of the algorithm.

2. In-depth analysis and optimization of the Adaptive Verification for Analogue Circuits algorithm.

   (a) Ablation study of the hyper-parameter tuning and algorithm adjustments. These include the study of different acquisition functions and optimizers for the gradient descent version of the algorithm.

   (b) Designed and implemented the Relative Value Error evaluation metric, used for evaluating the algorithm's performance on synthetic circuits.

3. Contributed to the design of the synthetic benchmark, used to validate the Adaptive Verification for Analogue Circuits algorithm.

4. Implemented the Specification Failure Probability risk metric, as an indicator to the possibility of specification failure for a circuit's response.

5. Implemented the Unequal (Dynamic) budget allocation, which assigns a different number of simulations for each of the circuit's responses with respect to the Specification Failure Probability metric.

8.3 List of original publications

1. Manolache Cristian, Alexandru Caranica, Marius Stănescu, Horia Cucu, Andi Buzo, Cristian Diaconu, and Georg Pelz. "Advanced operating conditions search applied in analog circuit verification." In 2022 18th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD), pp. 1-4. IEEE, 2022, DOI: 10.1109/SMACD55068.2022.9816209.

2. Manolache Cristian, Alexandru Caranica, Horia Cucu, Andi Buzo, Cristian Diaconu, and Georg Pelz. "Enhanced candidate selection algorithm for analog circuit verification." In 2022 International Semiconductor Conference (CAS), pp. 137-140. IEEE, 2022, DOI: 10.1109/CAS56377.2022.9934364.

3. Manolache Cristian, Cristina Andronache, Alexandru Caranica, Horia Cucu, Andi Buzo, Cristian Diaconu, and Georg Pelz. "Adaptive Planning Search Algorithm for Analog Circuit Verification." In 2023 International Semiconductor Conference (CAS), pp. 81-84. IEEE, 2023, DOI: 10.1109/CAS59036.2023.10303674.

4. Manolache Cristian, Cristina Maria Andronache, Alexandru Caranica, Horia Cucu, Andi Buzo, Cristian Vasile Diaconu, and Georg Pelz. "Applying Multi-objective Acquisition Function Ensemble for a candidate proposal algorithm." In 2023 International Conference on Speech Technology and Human-Computer Dialogue (SpeD), pp. 116-121. IEEE, 2023, DOI: 10.1109/SpeD59241.2023.10314884.

5. Manolache Cristian, Cristina Andronache, Alexandru Guzu, Alexandru Caranica, Horia Cucu, Andi Buzo, and Georg Pelz. "Synthetic Benchmark for Data-Driven Pre-Si Analogue Circuit Verification." Electronics 13, no. 13 (2024): 2600. ISI IF 2.6 (Q2), DOI: 10.3390/electronics13132600, WOS:001269706000001.

6. Manolache Cristian, Alexandru Guzu, Cristina Andronache, Alexandru Caranica, Horia Cucu, Gazmend Alia, Andi Buzo, and Georg Pelz. "SFP: A Metric for Assessing Circuit Failure Probability During Pre-Si Verification". UPB Scientific Buletin, Sec. C Electr. Eng. Comput. ISI IF 0.2 (Q4), WOS:001380943300014.

8.4 List of contracts

- Research Assistant (June 2021 – May 2022) at University Politehnica of Bucharest, in the project “Efficiency and Effectiveness of Circuit Verification using Machine Learning Methods v2”, R&D project funded by Infineon Technologies Romania & Co. SCS.

- Research Assistant (June 2022) at University Politehnica of Bucharest, in the project “Efficiency and Effectiveness of Circuit Verification using Machine Learning Methods v3”, R&D project funded by Infineon Technologies Romania & Co. SCS.

- Private scholarship from Infineon Technologies Romania & Co. SCS (July 2022 - December 2024).

- Research Assistant (Jan 2025 – May 2025) at POLITEHNICA Bucharest, in the project “Generative AI for Requirement Engineering (GenAI4RE)”, R&D project funded by Infineon Technologies Romania & Co. SCS, contract no. 4520242912.

8.5 Perspectives for further developments

The work presented in this thesis has resulted in a functional algorithm currently available for use within an industrial company. As the algorithm has been primarily tested on synthetic circuits and a limited number of real circuit models, feedback from its application to a wider range of real circuits would be valuable for further validation and improvement. A key area for future work is the further validation of the SFP metric. While initial results are promising, more extensive testing, particularly on the synthetic benchmark under various failable and non-failable scenarios, is needed to fully establish its reliability. Finally, although this thesis focuses on Pre-Silicon verification, the developed algorithm may be applicable to Post-Silicon verification as well. Both stages involve simulations under different OCCs, suggesting that the AVAC algorithm could be adapted for use in this subsequent phase of circuit verification.

References

[1] Balandat, M., Karrer, B., Jiang, D., Daulton, S., Letham, B., Wilson, A. G., and Bakshy, E. (2020). Botorch: A framework for efficient monte-carlo bayesian optimization. *Advances in neural information processing systems*, 33:21524–21538.

[2] Bergstra, J. and Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305.

[3] Cressie, N. (1989). Geostatistics. *The American Statistician*, 43(4):197–202.

[4] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197.

[5] Devarajegowda, K. and Ecker, W. (2018). Meta-model based automation of properties for pre-silicon verification. In *2018 IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC)*, pages 231–236. IEEE.

[6] Dobler, M., Harrant, M., Rafaila, M., Pelz, G., Rosenstiel, W., and Bogdan, M. (2015). Bordersearch: An adaptive identification of failure regions. In *2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1036–1041. IEEE.

[7] Farooq, U. and Mehrez, H. (2021). Pre-silicon verification using multi-fpga platforms: A review. *Journal of Electronic Testing*, 37(1):7–24.

[8] Garrido-Merchán, E. C. and Hernández-Lobato, D. (2020). Dealing with categorical and integer-valued variables in bayesian optimization with gaussian processes. *Neurocomputing*, 380:20–35.

[9] Gaur, P., Rout, S. S., and Deb, S. (2019). Efficient hardware verification using machine learning approach. In *2019 IEEE International Symposium on Smart Electronic Systems (iSES)(Formerly iNiS)*, pages 168–171. IEEE.

[10] Gielen, G., Xama, N., Ganesan, K., and Mitra, S. (2019). Review of methodologies for pre-and post-silicon analog verification in mixed-signal socs. In *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1006–1009. IEEE.

[11] Gupta, A., Ramanath, R., Shi, J., and Keerthi, S. S. (2021). Adam vs. sgd: Closing the generalization gap on image classification. In *OPT2021: 13th Annual Workshop on Optimization for Machine Learning*.

[12] He, C., Hu, H., and Li, P. (2021). Applications for machine learning in semiconductor manufacturing and test. In *2021 5th IEEE Electron Devices Technology & Manufacturing Conference (EDTM)*, pages 1–3. IEEE.

[13] Hu, H., Zheng, Q., Wang, Y., and Li, P. (2018). Hfmv: Hybridizing formal methods and machine learning for verification of analog and mixed-signal circuits. In *Proceedings of the 55th Annual Design Automation Conference*, pages 1–6.

[14] Khan, W., Azam, B., Shahid, N., Moeed Khan, A., and Shaheen, A. (2019). Formal verification of digital circuits using simulator with mathematical foundation. *Applied mechanics and materials*, 892:134–142.

[15] Knagg, O. (2019). An intuitive guide to gaussian processes. https://towardsdatascience.com/an-intuitive-guide-to-gaussian-processes-ec2f0b45c71d. Accessed: 2024-08-02.

[16] Kukkonen, S. and Lampinen, J. (2005a). Gde3: The third evolution step of generalized differential evolution. In *2005 IEEE congress on evolutionary computation*, volume 1, pages 443–450. IEEE.

[17] Kukkonen, S. and Lampinen, J. (2005b). GDE3: the third evolution step of generalized differential evolution. In *2005 IEEE Congress on Evolutionary Computation*, volume 1, pages 443–450 Vol.1. ieeexplore.ieee.org.

[18] Lim, H. G., Jang, J., Ju, B. K., Ko, J. W., and Kim, C. O. (2023). A test vector selection method based on machine learning for efficient presilicon verification. *Expert Systems with Applications*, 224:120056.

[19] Lyu, W., Yang, F., Yan, C., Zhou, D., and Zeng, X. (2018). Batch bayesian optimization via multi-objective acquisition ensemble for automated analog circuit design. In *International conference on machine learning*, pages 3306–3314. PMLR.

[20] McKay, M. D., Beckman, R. J., and Conover, W. J. (2000). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 42(1):55–61.

[21] Montgomery, D. (2004). *Design and Analysis of Experiments (pp. 71–198)*. John wiley & sons.

[22] Owen, A. B. (1992). Orthogonal arrays for computer experiments, integration and visualization. *Statistica Sinica*, pages 439–452.

[23] Owen, A. B. (2013). *Monte Carlo theory, methods and examples*. Stanford.

[24] Pronzato, L. and Müller, W. G. (2012). Design of computer experiments: space filling and beyond. *Stat. Comput.*, 22(3):681–701.

[25] Robič, T. and Filipič, B. (2005). Differential evolution for multiobjective optimization. In *International conference on evolutionary multi-criterion optimization*, pages 520–533. Springer.

[26] Şandru, E.-D., David, E., and Pelz, G. (2019). Pre-silicon yield estimation using machine learning regression. In *2019 26th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, pages 103–104. IEEE.

[27] Stănescu, M., Vişan, C., Sandu, G., Cucu, H., Diaconu, C., Buzo, A., and Pelz, G. (2021). Multi-objective optimization algorithms for automated circuit sizing of analog/mixed-signal circuits. In *2021 International Semiconductor Conference (CAS)*, pages 117–120. IEEE.

[28] Visan, C. (2024). *Artificial Intelligence Techniques For Integrated Circuit Design Automation*. PhD thesis, National University of Science and Technology Politehnica of Bucharest, Bucharest, Romania.

[29] Vişan, C., Pascu, O., Stănescu, M., Cucu, H., Diaconu, C., Buzo, A., and Pelz, G. (2021). Versatility and population diversity of evolutionary algorithms in automated circuit sizing applications. In *2021 International Conference on Speech Technology and Human-Computer Dialogue (SpeD)*, pages 68–73. IEEE.

[30] Zhang, S., Yang, F., Yan, C., Zhou, D., and Zeng, X. (2021). An efficient batch-constrained bayesian optimization approach for analog circuit synthesis via multi-objective acquisition ensemble. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 41(1):1–14.

[31] Zhou, P., Feng, J., Ma, C., Xiong, C., Hoi, S. C. H., et al. (2020). Towards theoretically understanding why sgd generalizes better than adam in deep learning. *Advances in Neural Information Processing Systems*, 33:21285–21296.

[32] Zhuo, C., Yu, B., and Gao, D. (2017). Accelerating chip design with machine learning: From pre-silicon to post-silicon. In *2017 30th IEEE International System-on-Chip Conference (SOCC)*, pages 227–232. IEEE.